Collana diretta da Giulio Palumbo

6

Claudio Mantovani

# Il Computer in Casa

CLUB DEL LIBRO FRATELLI MELITA

# INDICE

# INTRODUZIONE

Perché scrivere un libro sui programmi? I motivi possono essere diversi. Innanzitutto il piacere innegabile nello scrivere esiste sempre, qualsiasi sia l'argomento trattato.

La scelta dell'argomento invece è dettata dalla più grossa rivoluzione a cui stiamo assistendo: la rivoluzione informatica.

Non riusciamo nemmeno a comprendere il suo intimo significato: infatti compriamo gli elaboratori (dato che la moda ci dice di farlo) e poi li lasciamo lì o al massimo ci giochiamo con dei giochi già in commercio. Non costruiamo invece gli strumenti (i programmi), che ci consentirebbero di usare in maniera intelligente il nostro nuovo elaboratore.

Perché chi compera l'elaboratore (che da questo momento chiamerò Sig. Vip), non costruisce gli strumenti a lui utili?

Questo avviene per due motivi principali. Il primo di essi è che il Sig. Vip è prigro!

Perciò si chiede: perché sforzarsi ad imparare delle cose (apparentemente) complicate, se in commercio esiste già quello che io desidero e di cui ho bisogno?

La pigrizia è una cosa seria: anch'io sono fondamentalmente pigro. Però questo dolce stato di pigrizia si può facilmente superare. In quale modo? Pensando a ciò che si potrebbe fare se non fossimo pigri, alle cose che potremmo costruire e a come ci potremmo sentire realizzati dopo aver tradotto in realtà ciò che abbiamo progettato.

Il secondo motivo per cui il Sig. Vip compera i programmi a lui necessari è da addebitare alla stampa in genere. In primo luogo alla grande pubblicità che viene fatta per il *software* (*) e al suo basso costo (ma molto spesso il basso costo corrisponde ad infima qualità). In

(*) Parola inglese che sta a significare tutte le parti non materiali di un qualsiasi elaboratore.

secondo luogo, invece, alla inadeguata campagna per la diffusione della semplicità della programmazione.

Orbene questo libro ha la pretesa di insegnare al Sig. Vip che programmare non è poi così difficile e che perciò il software si può benissimo sviluppare (o adattare) in casa propria, con una soddisfazione davvero inimmaginabile.

Il testo si svilupperà in questo modo: dopo un primo capitolo di introduzione riguardante la descrizione ed una breve storia degli elaboratori, vi sarà un capitolo che descriverà il software e la maniera di produrlo in modo autonomo. Quindi vi saranno quattro capitoli riguardanti esempi di programmazione. Infine l'ultimo capitolo tratterà l'adattamento di programmi scritti per altri elaboratori, a quello, che il Sig. Vip possiede.

# 1. INTRODUZIONE AGLI ELABORATORI

## 1.1 BREVE STORIA DEGLI ELABORATORI ELETTRONICI

Da quando l'uomo ha iniziato a fare dei calcoli, ha sentito la necessità di costruirsi qualcosa che lo aiutasse in questa noiosa e ripetitiva operazione.

Il primo meccanismo usato per facilitare il calcolo manuale fu l'abaco, già noto agli antichi cinesi e babilonesi, e poi ripreso dai greci. Il primo calcolatore numerico meccanico, invece, fu costruito da Pascal e Leibnitz nel diciassettesimo secolo. Nel 1812 Charles Babbage, un matematico britannico, progettò una macchina che chiamò «Difference Engine» e che era d'aiuto nel calcolo di tavole matematiche. Più tardi nel 1833, lo stesso Babbage progettò una «Analytical Engine» che avrebbe dovuto essere completamente automatica, ma che non fu mai realizzata per l'impossibilità di costruire le parti meccaniche necessarie con sufficiente precisione.

La serie di operazioni utili a risolvere un problema doveva essere fornita alla macchina tramite schede perforate e la memoria avrebbe potuto contenere 1000 numeri di 50 cifre ciascuno. Vi sarebbe stata insita anche la possibilità di modificare il corso dei calcoli in base ai risultati parziali, tramite schede di salto. I principi dell'«Analytical Engine» costituiscono ancora la base dei progetti di molti calcolatori attuali.

Nel 1937 il Prof. Howard Aiken ideò un calcolatore elettromeccanico, completamente automatico, che utilizzava i principi della macchina di Babbage. Questo calcolatore, conosciuto con il nome di Mark I, fu completato nel 1944 e poteva eseguire una qualunque sequenza di calcoli, consistente nelle quattro operazioni fondamentali aritmetiche, e con la possibilità di modificare il corso dei calcoli in base ai risultati parziali ottenuti. Le informazioni venivano immesse per mezzo di schede perforate ed azionando opportuni interruttori, i risultati venivano emessi su scheda o scritti su telescrivente. È da notare che una moltiplicazione richiedeva circa 3 secondi.

I calcolatori elettronici non potevano comunque essere costruiti prima del 1950: questo per un motivo molto semplice, e cioè per il fatto che i componenti elettronici necessari alla loro costruzione non erano ancora disponibili materialmente. Comunque, ormai, i concetti di base erano assodati da oltre un secolo.

Perciò nel 1946 J.P. Eckert e J.W. Manchley poterono affrontare il problema del trattamento rapido di una quantità crescente di dati, relativi a studi metereologici e balistici, per i quali era necessaria un'alta velocità di elaborazione. Essi pensarono che una tale esigenza poteva essere soddisfatta soltanto tramite tecniche elettroniche e così progettarono l'ENIAC (Electronic Numerical Integrator and Computer). Questa storica macchina conteneva 18.000 valvole e costituiva un gran passo in avanti nella tecnologia dei calcolatori non avendo alcuna parte mobile interna. I numeri erano di 10 cifre decimali, l'addizione richiedeva 0,2 millisecondi e la moltiplicazione 2,8 millisecondi; vi era un solo particolare a suo discapito: si rilevava un guasto ogni 10 minuti!!!

Un programma veniva imposto al calcolatore tramite collegamenti di fili elettrici (tale azione in gergo si chiama cablaggio); essendo, per forza di cose, quest'operazione piuttosto lenta, l'utilità dell'ENIAC era di conseguenza limitata.

Il calcolatore elettronico IBM 605 fu il primo ad essere costruito in un certo numero di esemplari (4.000 tra il 1946 ed il 1960).

Nell'IBM Card — Programmer — Calculator (CPC) veniva aggiunta al 605 la possibilità di memorizzare un programma su schede: sequenze alternative di schede venivano usate a seconda dei risultati parziali e questo rendeva il CPC una macchina versatile ed adattabile a diversi tipi di problemi.

Nel 1945 Von Neumann avanzò le proposte di un tipo di calcolatore in cui il programma fosse memorizzato all'interno di esso. La memorizzazione interna avrebbe permesso alla macchina di ottenere quella flessibilità che l'avrebbe resa vincente negli anni seguenti. L'idea di Von Neumann fu realizzata nel 1950 con l'EDVAC (Electronic Discret Variabile Automatic Computer); venivano impiegate 12 differenti istruzioni, l'addizione richiedeva 0,9 millisecondi e la moltiplicazione circa 2,9 millisecondi.

Sempre nel 1945 i laboratori del M.I.T. affrontarono lo studio di un simulatore di volo e nel 1951 costruirono, come risultato dei loro sforzi, il Whirlwind I. Esso conteneva 5000 valvole e 11.000 diodi a semiconduttore. Le velocità di esecuzione erano paragonabili a quelle delle macchine odierne: 3 microsecondi per l'addizione e 16 microsecon-

di per la moltiplicazione. Il programma risiedeva nella memoria della macchina; esistevano 27 istruzioni e 2048 celle di memoria. La macchina era estremamente affidabile e molti concetti introdotti con essa sono ancor oggi utilizzati.

Dal 1951 ad oggi la tecnologia ha fatto passi da gigante e ha subìto uno sviluppo eccezionalmente rapido. Si è passati dalle valvole, ai semiconduttori e ai circuiti integrati nel giro di 30 anni. Le macchine che oggi si possono costruire sono molto più potenti di quelle che si potevano immaginare agli inizi dell'elettronica ed il futuro è ancora al di fuori di qualsiasi previsione. Se si paragona lo sviluppo dell'elettronica a quello della carta stampata (sviluppi che costituiscono due rivolte culturali essenziali per l'uomo moderno) si può vedere che, mentre per la stampa ci sono voluti 400 anni per la diffusione del suo uso, per l'elettronica ce ne sono voluti soltanto 30! Da questo semplice paragone si può intuire che vi saranno problemi sociali ed educativi nel campo dell'elettronica, in genere. In effetti l'uomo ha avuto tutto il tempo necessario per adattarsi alla stampa, ai giornali e ai libri visti i suoi tempi di sviluppo. Mentre invece per quel che riguarda la nostra elettronica si deve notare che questo è avvenuto molto rapidamente: mezza generazione in rapporto alle 5 o 6 della stampa! Perciò in che modo potrà adattarsi il Sig. Vip a questa nuova cultura e a questi nuovi strumenti? È un dibattito ancora aperto e avrà bisogno forse di una cosa, in particolare: cioè che l'uomo abbia una grossa apertura mentale e non si fossilizzi su cose particolari, su strumenti obsoleti, ma rimanga sempre aperto a tutti i progressi di questa nuova branca della scienza umana.

In aggiunta a tutto ciò si può ulteriormente osservare come gli sviluppi futuri della microelettronica non siano assolutamente prevedibili, in quanto i progressi di essa sono difficilmente tempificabili e controllabili.

## 1.2 STRUTTURA DI UN ELABORATORE

Per conoscere un poco più a fondo cos'è un elaboratore si può fare riferimento a molti schemi e a svariati concetti. Lo schema logico che ho scelto e a cui mi collegherò è il seguente:

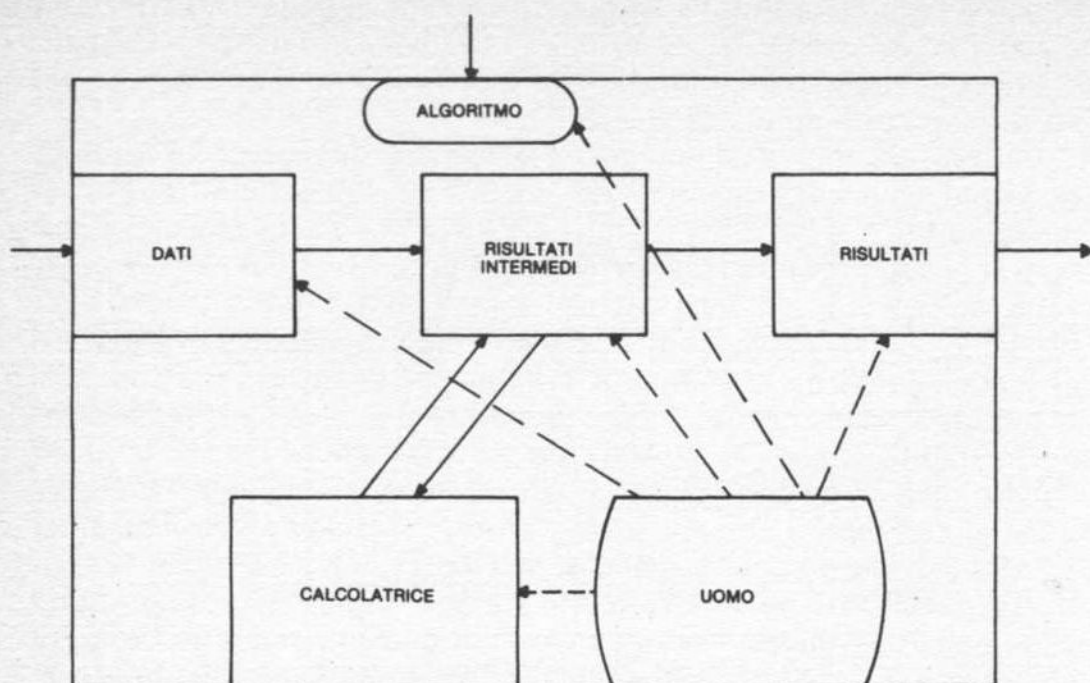



**Fig. 1**

Come si vede l'uomo ha rapporti con tutte le altre parti dello schema governandole e dirigendole a seconda dei suoi scopi. D'altro canto vi è un flusso continuo di dati, trattati da un algoritmo, che entrano nel sistema, vengono manipolati e ne escono come risultati finali.

Da questi semplici concetti deriva quella che è la struttura *hardware* (*) di un elaboratore. In questo secondo, e più realistico schema, le parti logiche del sistema di *Fig. 1* sono sostituite da componenti fisiche dell'elaboratore vero e proprio.

(*) Hardware: parola inglese che significa tutto ciò che vi è di tangibile (fisicamente) in un sistema elaborativo.

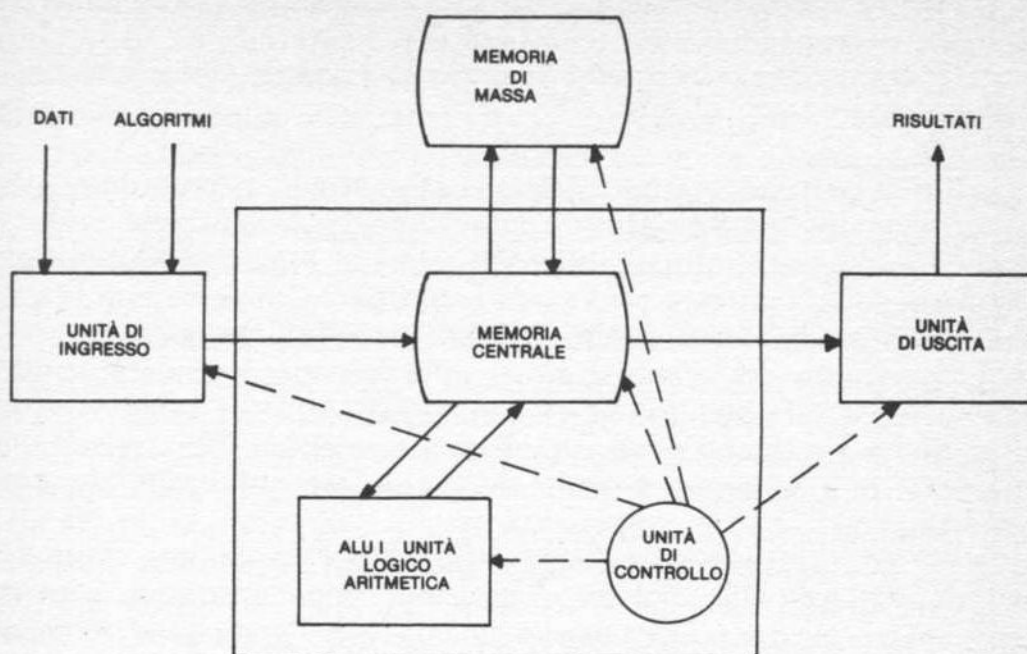



Fig. 2

Possiamo fare subito alcune utili osservazioni confrontando lo schema logico di *Fig. 1* con quello fisico di *Fig. 2*. Si nota subito che il grosso quadrato al centro della figura è rimasto invariato: sono cambiati, però, i suoi contenuti. Il grosso quadrato sarà, d'ora in poi, chiamato CPU (Central Process Unit) (*): è il cuore dell'elaboratore.

In esso troviamo per prima cosa, l'unità di controllo, che ha sostituito l'uomo dello schema logico e che ha il compito principale di governare tutto ciò che accade all'interno dell'elaboratore. I compiti specifici dell'unità di controllo sono, comunque, i seguenti:

— dovrà leggere, interpretare e fare eseguire le istruzioni dei programmi degli utenti (dovrà curare i salti, le decisioni, etc.);

— dovrà regolare, tempificare e sincronizzare le azioni delle unità che verranno via via attivate;

(*) In italiano si può chiamare unità centrale di processo.

— dovrà regolare gli scambi dei dati tra la memoria centrale, la memoria di massa e l'ALU (Unità Logica Aritmetica);
— dovrà utilizzare e regolare le porte di ingresso e di uscita per i dati e gli algoritmi.

L'ALU (o unità logica aritmetica) sostituisce la calcolatrice dello schema logico. Come dice la sua definizione essa è utilizzata dall'unità di controllo per effettuare tutti i tipi di calcoli logico aritmetici. Perciò dovrà essere attivata dall'unità di controllo e, per effettuare i calcoli, dovrà essere a conoscenza degli operandi su cui operare.

Questa parte dell'elaboratore è, relativamente parlando, molto semplice perché è costituita da alcuni registri (piccole memorie) e da circuiti che effettuano poche operazioni elementari. Chiaramente la sua potenza è superiore a quella delle calcolatrici portatili che ben conosciamo.

Le memorie, invece, sono i magazzini in cui l'elaboratore memorizza tutti i dati e gli algoritmi che gli servono, sia per effettuare lavori in tempo reale, sia per portare a termine dei lavori che gli saranno imposti in futuro o che gli sono stati imposti in passato (ma che non necessitano di risposte immediate). La differenza tra memoria centrale e memoria di massa è che, fisicamente, la memoria centrale risiede all'interno della CPU, mentre la memoria di massa può essere al di fuori della CPU stessa. La differenza funzionale, invece, risiede nel fatto che la CPU lavora solo con la memoria centrale, mentre la memoria di massa viene utilizzata dall'unità di controllo per memorizzare dati ed algoritmi, non utili al momento. Infine, la memoria centrale ha una limitata dimensione, ma possiede un accesso di dati molto rapido, al contrario della memoria di massa che ha queste caratteristiche invertite.

Qualche altra considerazione si può fare a proposito delle unità di ingresso e di uscita. Tali unità servono, come dice il nome, per immettere i dati, gli algoritmi ed i comandi alla CPU e poi per visualizzare i risultati. Le unità di ingresso si possono identificare con la tastiera degli *home computer* (*): essa è simile ad una macchina da scrivere con cui si possono digitare i comandi che saranno visualizzati sul video (**).

(*) Parola inglese che significa elaboratore personale o per la casa.

(**) Per altri tipi di elaboratori, più grossi e potenti degli home computer, vi potranno essere svariate unità di ingresso, come telescriventi, lettori di schede, lettori di nastri, etc.

Le unità di uscita si possono invece identificare con il vostro video oppure con i nastri o i dischetti del vostro home computer. Su queste unità fisiche finiranno i dati che vorrete leggere o visualizzare per controllare i risultati dei calcoli da voi implementati.

Da questi brevi cenni si può comprendere come un elaboratore sia essenzialmente una macchina semplice da capire, in quanto fa riferimento a degli schemi logici ormai consolidati nella mente umana. Tali schemi, che per gli home computer sono semplici, possono diventare molto più complicati se si prendono in considerazione gli elaboratori usati per compiti più complessi.

Comunque bisogna sempre tenere presente che lo schema base di un qualsiasi elaboratore farà in ogni caso riferimento a quello di *Fig. 2*, potendo aumentare le varie unità, espandere le memorie o potenziare la CPU.

In effetti, si potrebbero modificare le unità di ingresso, facendole diventare delle telescriventi o dei video dedicati; cambiare le unità di uscita in video, video-grafici o plotter. Potrebbero anche cambiare le memorie di massa, diventando quei grossi nastri o grossi dischi in cui vengono contenute milioni di parole. Oppure la stessa CPU, diventando quella potentissima unità di controllo usata per il lancio di oggetti nello spazio o per altri simili compiti certamente non alla portata di una sola mente umana.

Però in ogni caso e sistema elaborativo potrete ritrovare tutte quelle parti o funzioni che sono contenute nel vostro piccolo home computer.

Perciò, una volta compreso lo schema di base di un elaboratore, non dovreste fare molta fatica a capire la logica di qualsiasi altro sistema elaborativo, grosso o piccolo che sia.

## 1.3 CLASSIFICAZIONE DEGLI ELABORATORI

In questo paragrafo intendiamo analizzare quali tipi di sistemi elaborativi esistono in commercio.

Come abbiamo già accennato in precedenza, non tutti gli elaboratori sono «uguali». Perciò si possono analizzare parametri abbastanza diversi per fare una sorta di classificazione tra di essi.

Ad esempio si può considerare:

— il costo, che può variare dalle 100.000 lire al miliardo;

— la tecnologia, che si può suddividere in circuiti a componenti discreti (ora sempre meno usati), circuiti LSI (a larga scala di integrazio-

ne) o circuiti VLSI (a larghissima scala di integrazione, tra cui sono contenuti i microprocessori);

— le applicazioni per cui si utilizza l'elaboratore. Si può ad esempio avere un elaboratore che è in grado di compiere una sola funzione, anche se complessa (elaboratore dedicato), o uno che può compiere un gran numero di funzioni alternative ed anche più funzioni contemporaneamente.

Una classificazione molto comune, perché sottolinea alcuni aspetti importanti di ciascuna classe, è quella che sembra fatta tenendo conto delle «dimensioni» degli elaboratori: si possono perciò individuare gli home computer, i *personal computer* (*) i minicalcolatori ed i maxicalcolatori (o *main frames*).

Pur non essendo tale classificazione per nulla rigida ed essendo assai spesso difficile collocare una specifica macchina in una delle quattro classi, esistono delle caratteristiche ben precise che vogliamo mettere in luce.

### 1.3.1 GLI HOME COMPUTER ED I PERSONAL COMPUTER

Questo tipo di elaboratore è quello che ci interessa più da vicino, vista la diffusione che ha raggiunto in così poco tempo.

Un home computer (**) è un sistema variamente organizzato, comprendente:

— un microprocessore, ossia un singolo elemento logico della tecnologia LSI o VLSI, che rappresenta il cuore dell'elaboratore;

— una memoria RAM (cioè ad accesso casuale e con un tempo di accesso ai dati pressoché costante), di dimensioni ridotte (8-16 Kbyte (***) (****));

(*) Parola inglese che significa calcolatore personale e che dovrebbe aiutare la singola persona durante lo svolgimento della sua giornata di lavoro o non.

(**) Da questo momento i concetti espressi per l'home computer valgono anche per i personal computer tranne alcune particolarità che saranno evidenziate.

(***) Il bit è l'elemento più semplice d'informazione.

Ad esempio, alla domanda: l'oro è più prezioso dell'argento? Il messaggio di risposta contiene una singola informazione: un *sì* o un *no*. La risposta sì o no in questo caso rappresenta un bit. Un insieme di 8 bit formano 1 byte. Invece 1 Kbyte rappresenta $2^{10}$ byte, cioè 1024 byte o 8192 singoli bit.

(****) Per il personal computer la memoria RAM sarà anche più estesa e può raggiungere i 128 Kbyte.

— una memoria ROM (memoria a sola lettura), da cui è possibile leggere dati o programmi, ma in cui non è possibile scrivere. In essa, in genere, vengono scritte una volta per tutte le operazioni tipiche delle applicazioni per cui l'home computer viene impiegato. Si parla spesso di memoria microprogrammata, facendo riferimento a quella tecnica che permette di trasformare una codifica (cioè un programma e quindi del software), in una specifica struttura della macchina (in qualcosa di hardware). In questo modo è possibile ottimizzare la velocità di esecuzione dell'elaboratore, permettendogli di ottenere migliori prestazioni;

— delle interfacce per le operazioni di ingresso-uscita.

Come abbiamo già visto possono esistere un gran numero di unità periferiche utilizzate per le operazioni di ingresso-uscita. È chiaro anche che esiste una grande differenza tra di esse. Diverse sono le velocità, diverso è il loro modo di operare rispetto al linguaggio, proprio, dell'elaboratore, etc.

Quindi, per poter collegare ad un elaboratore una specifica unità periferica è necessario interporre un'apparecchiatura, detta interfaccia, che ha, da un lato, il compito di rendere comprensibile all'elaboratore il linguaggio della periferica, dall'altro quello di permettere a questa di parlare con esso. Compito delle interfacce è anche quello di controllare che il funzionamento della periferica, e quindi la trasmissione e ricezione dei dati, sia corretto, che non si siano verificati errori e che l'operazione di trasferimento sia terminata. I risultati di questi controlli, sotto forma di segnali e di *flag* (*) vengono poi elaborati dall'unità centrale che, come si è visto, ha tra i suoi compiti, quello di presiedere (**) all'esecuzione di operazioni di ingresso-uscita.

Il mercato attuale degli home computer e dei personal computer è estremamente vasto ed in rapida evoluzione. La qualità raggiunta è

(*) Un *flag* è come dice la sua traduzione in italiano, una bandiera: se c'è (vale a dire se ha un certo valore) significa che è successo qualcosa nel programma; se non c'è, ancora non è successo niente.

(**) Per il personal computer, come componenti base, dovremo citare anche le memorie di massa costituite da dischetti (dischi di materiale ferromagnetico deformabile larghi una decina di centimetri), o da piccoli winchester o dischi rigidi (di materiale ferromagnetico indeformabile e leggermente più grossi dei dischetti). Le informazioni che si possono memorizzare in questi strumenti, sono moltissime se paragonate a quelle contenute nella memoria RAM. Infatti si parte dai 256 Kbyte dei dischetti, per arrivare fino all'ordine dei MByte, per quel che riguarda i winchester.

abbastanza uniforme e, a parità di prezzi, esistono solo lievi differenze tra le case costruttrici.

La flessibilità e la potenza di queste macchine rende molto difficile fare una rigida distinzione tra home computer, personal computer ed anche minicalcolatori.

Comunque le loro caratteristiche portano a pensare che tali dispositivi possano essere impiegati per qualsiasi tipo di applicazione, partendo dall'uso molto semplice che se ne fa in casa ed arrivando a quelli molto sofisticati richiesti sul campo del lavoro.

Le peculiarità fondamentali di questo tipo di elaboratori sono il basso costo, le limitate dimensioni, il basso consumo energetico e l'estrema modularità della loro architettura.

Il software utilizzato dagli home e dai personal computer, è già molto sviluppato sia dalle stesse case produttrici degli elaboratori, che da società appositamente create. Tuttavia, la facilità della loro comprensione e gestione, rende possibile la loro programmazione anche da parte del Sig. Vip. È quello che cercheremo di sviscerare dall'inizio del prossimo capitolo.

### 1.3.2 I MINICALCOLATORI

I minicalcolatori si possono identificare come degli elaboratori più potenti dei personal computer, ma che non hanno tutte le prestazioni dei maxicalcolatori.

La caratteristica fondamentale, che li contraddistingue e li diversifica dai personal ed home computer, è la possibilità di poter avere contemporaneamente, più di un utente che si serve della CPU. Questa peculiarità si chiama multiprogrammazione. Ciò vuol dire che, nello stesso istante, più utenti possono richiedere all'elaboratore delle prestazioni diverse. La CPU riesce a gestire le differenti richieste sia dando delle priorità ai programmi, sia dedicando degli strumenti ad un utente e strumenti diversi ad un altro utente. Di solito, per fare un esempio, se un utente chiede un output di dati ed un altro chiede l'esecuzione di operazioni logiche, l'unità di controllo assegnerà al primo l'unità di uscita e al secondo l'ALU. Chiaramente l'unità di controllo, in questo caso, dovrà essere molto più elaborata di quella dei semplici home computer ed in più vi saranno dei programmi dedicati per questa ed altre utilità che risiederanno stabilmente nella memoria centrale. Da ciò deriva una lievitazione della dimensione della memoria centrale.

In aggiunta, si può notare che i minicalcolatori sono ormai diventati di uso comune nel campo del lavoro, considerato il rapporto tra le loro prestazioni (che si possono tranquillamente paragonare a quelle di un elaboratore di dimensioni molto più estese) ed il loro costo, che non sarà mai elevatissimo.

### 1.3.3 I MAXICALCOLATORI

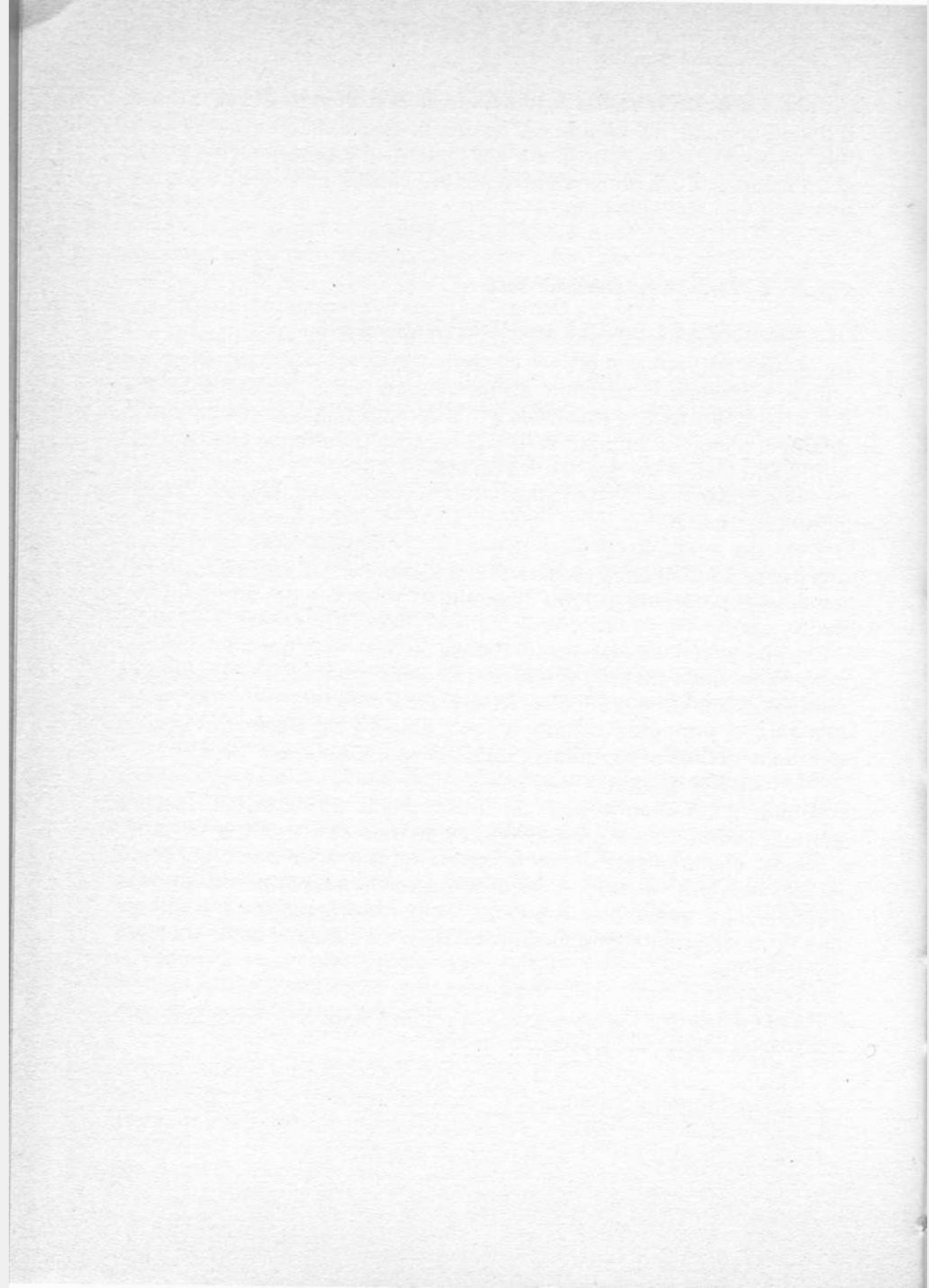
Il termine maxicalcolatore (o main frame) sta qui ad indicare, non tanto un elaboratore di grosse dimensioni, bensì piuttosto una macchina utilizzabile in parallelo da un numero molto elevato di utenti, con programmi diversi tra di loro e con diverse esigenze per quanto riguarda il tempo di risposta. A tutti l'elaboratore fornisce una risposta sufficientemente rapida ed offre un'alta disponibilità di memoria, sia centrale che di massa, cosicché l'utente è convinto di avere a disposizione tutto il sistema.

L'utente, in questo caso, non deve preoccuparsi delle caratteristiche hardware della macchina e può dedicarsi ad utilizzare un grosso insieme di programmi molto diversificati tra loro e già contenuti nel main frame.

Un tale sistema è spesso composto da due unità centrali che operano in parallelo tra loro. Quest'ultima caratteristica, insieme all'alto numero di memorie di massa e di unità periferiche, è importante anche perché aumenta l'affidabilità complessiva del sistema in quanto può così far fronte ad indisponibilità di periferiche o di CPU.

Per chiudere questa breve rassegna sui vari tipi di elaboratore oggi esistenti in commercio, voglio sottolineare come sia difficile catalogare un elaboratore ed inserirlo in un gruppo invece che in un altro.

Si può anche osservare come i personal computer odierni abbiano le caratteristiche di potenza simili a quelle di un elaboratore di medie dimensioni di una decina di anni fa. E ciò, ancora una volta, evidenzia quanto lo sviluppo dell'elettronica e della microelettronica sia stato rapidissimo.

Per contro la sua evoluzione è ancora ai primi passi e le previsioni fatte per il futuro sicuramente saranno superate dal progresso di questa nuova branca della scienza umana.

# 2. DIAGRAMMI DI FLUSSO E CODIFICA

## 2.1 COSA VUOL DIRE PROGRAMMARE?

Sinora abbiamo analizzato soltanto la struttura esterna dell'elaboratore del Sig. Vip. Ma cosa c'è all'interno di esso? Quali compiti può assolvere? In che modo li esegue?

In questo paragrafo cercheremo di sviscerare questi problemi e di entrare più approfonditamente nel regno che si chiama software, ma che noi d'ora in poi, chiameremo più modestamente programmazione.

Per capire come un elaboratore possa eseguire le sequenze di comandi che gli daremo in pasto, bisogna pensare a lui come ad un esecutore, cieco e senza desideri di rivolta. Eseguirà qualsiasi cosa gli si chieda, a patto che gliela si chieda nella forma appropriata e sintatticamente corretta, senza protestare. Da ciò deriva il concetto che un elaboratore non sbaglierà mai, in ogni suo compito. Chiaramente se avrà dei problemi evidenti (come la rottura di qualcosa o come dei risultati inaspettati), bisognerà prima porre riparo ad essi e poi far eseguire all'elaboratore quello che noi vogliamo.

Torniamo alle sequenze di comandi. Quando abbiamo un problema da risolvere possiamo agire in tre diverse maniere: o ce lo risolviamo noi oppure lo diamo ad un nostro amico oppure, ancora, lo risolviamo tramite elaboratore. Nel primo caso non avremo alcuna difficoltà a comunicare i termini del problema, in quanto ci facciamo tutto in casa nostra. Negli altri due casi, invece, avremo alcune complicazioni nel dover esporre il problema al risolutore, sia esso l'elaboratore o il nostro amico.

Cominciamo col supporre di delegare un nostro amico per la risoluzione di un problema specifico.

Ad ogni problema è sempre associata una descrizione, espressa, in genere, in linguaggio naturale, ovvero in altri tipi di linguaggio, come ad esempio il linguaggio matematico. Ciò che è necessario tener pre-

sente, a questo punto, è la sua espressione in un linguaggio (intrinseca alla descrizione di un problema).

Si possono notare anche i seguenti punti:

— la descrizione di un problema, in genere, non fornisce il metodo per una sua soluzione, ossia per il calcolo del risultato a partire dai dati di ingresso;

— la descrizione di un problema è talvolta ambigua ed imprecisa, nel senso che ad essa si possono associare molteplici interpretazioni;

— non è nemmeno detto, allo stato attuale delle conoscenze, che sia possibile calcolare una soluzione del problema;

— dati di ingresso e risultati in uscita possono essere di natura molto generale.

Torniamo al nostro problema.

Immaginiamo ora di avere un amico completamente privo di creatività, vale a dire un amico a cui possiamo dire: esegui questi comandi. E lui, senza protestare, porterà a termine il compito che gli abbiamo assegnato. È chiaro che per ottenere i risultati desiderati dal nostro amico, dovremo fornirgli una serie di regole e di istruzioni, espresse in linguaggio naturale, che gli permetteranno appunto, se eseguite, di arrivare alla soluzione del problema a partire dai dati in ingresso.

Tale insieme di regole per la soluzione di un problema in un numero finito di passi viene detto algoritmo. Dopo aver trovato il modo di assegnare l'esecuzione di un compito ad una persona, cioè al nostro amico, vediamo come possiamo assegnare l'esecuzione dello stesso compito al nostro elaboratore.

Se avete prestato attenzione, il nostro amico senza creatività è simile per quel che riguarda la soluzione di problemi, all'elaboratore.

Infatti esso può essere associato ad un automa, poiché porta a termine senza protestare ogni compito. Per contro l'elaboratore è di per sé un automa e quindi compie le stesse azioni del nostro amico. Perciò la maniera per comunicare con l'elaboratore è quella di costruire ed immettergli degli algoritmi di soluzione. Esso non farà altro che eseguire, in maniera pedestre, ma nello stesso tempo infinitamente precisa, tutte le nostre istruzioni.

A questo punto sorge un ulteriore problema: come comunicare con l'elaboratore? Chiaramente non in linguaggio italiano?

L'elaboratore avrà insita, come abbiamo visto nel primo capitolo, la possibilità di interpretare direttamente delle istruzioni scritte in un ben preciso linguaggio. Il linguaggio comprensibile ad un elaboratore

si chiama linguaggio macchina, che in effetti sarebbe l'insieme di istruzioni direttamente interpretabile da una specifica macchina.

Quindi tramite il linguaggio macchina possiamo trasferire all'elaboratore l'algoritmo che ci occorre. L'algoritmo si può anche chiamare programma: infatti un programma non è altro che l'insieme delle istruzioni che dovrà seguire l'elaboratore per risolvere un problema, e l'algoritmo è il procedimento di calcolo da seguire.

A questo punto fermiamoci un attimo. Cosa significa delegare la soluzione di un problema ad un elaboratore? Significa effettuare queste due operazioni:

1. *Individuare un algoritmo di soluzione del problema.*
2. *Esprimere l'algoritmo in linguaggio macchina.*

Per quanto riguarda il primo passo non dovremmo trovare ostacoli; almeno per quei problemi che hanno una possibile soluzione. Per il secondo, invece, li incontreremo sicuramente. Perché? Per un motivo molto semplice: il linguaggio macchina è una sequenza di uni e zeri che potrebbero spaventare, e a ragione, il Sig. Vip. In effetti esso rimane molto difficile da interpretare, o almeno molto noioso.

Per questa ragione nacque l'idea di costruire altri linguaggi chiaramente più potenti e meno complicati del linguaggio macchina, ma nello stesso tempo più vicini a quello con cui l'uomo tende ad esprimere gli algoritmi, ossia il linguaggio naturale.

È evidente che l'uso di tali linguaggi, detti ad alto livello, perché appunto lontani dal linguaggio macchina e più vicini a quello naturale, pone un nuovo problema: come mettere l'elaboratore in condizione di capire queste nuove espressioni?

Prima di rispondere a tale quesito vediamo il significato della definizione di linguaggio.

Un linguaggio è un insieme fissato di simboli e regole che governano i modi in cui i simboli possono essere combinati per ottenere delle comunicazioni comprensive ad un ascoltatore.

Il linguaggio naturale è il linguaggio con cui il Sig. Vip parla e scrive.

Un linguaggio artificiale è un linguaggio inventato per qualche scopo particolare.

Da ciò deriva che un linguaggio di programmazione è un linguaggio artificiale specificatamente progettato per esprimere programmi.

Infine analizziamo due ultime definizioni:

— la sintassi di un linguaggio è l'insieme di regole per la formazione di costrutti accettati da quel linguaggio, senza riferimento al loro significato;

— la semantica è l'insieme di regole che interpretano e danno un significato ai costrutti di quel particolare linguaggio.

Dopo questa lunga, e forse noiosa elencazione, possiamo rispondere al nostro quesito. Fissata, in maniera non ambigua, la sintassi e la semantica di un linguaggio programmativo ad alto livello, il problema di far comprendere all'elaboratore tale linguaggio può essere risolto tramite i seguenti passi:

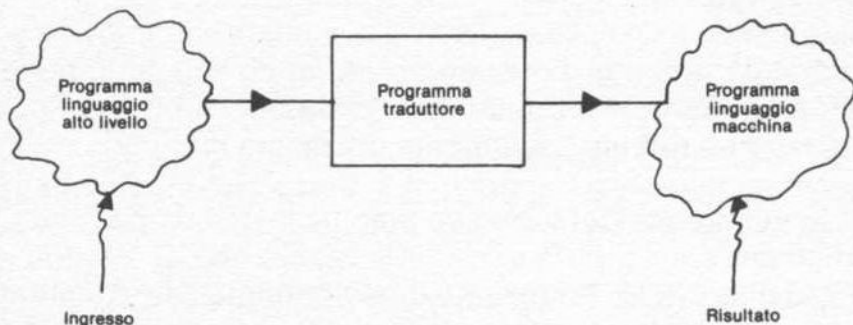
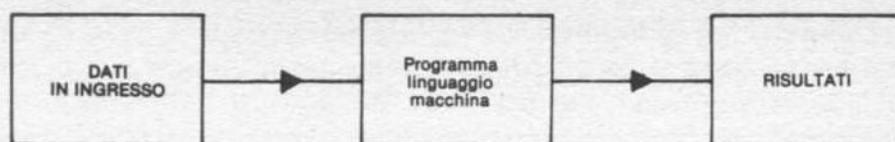
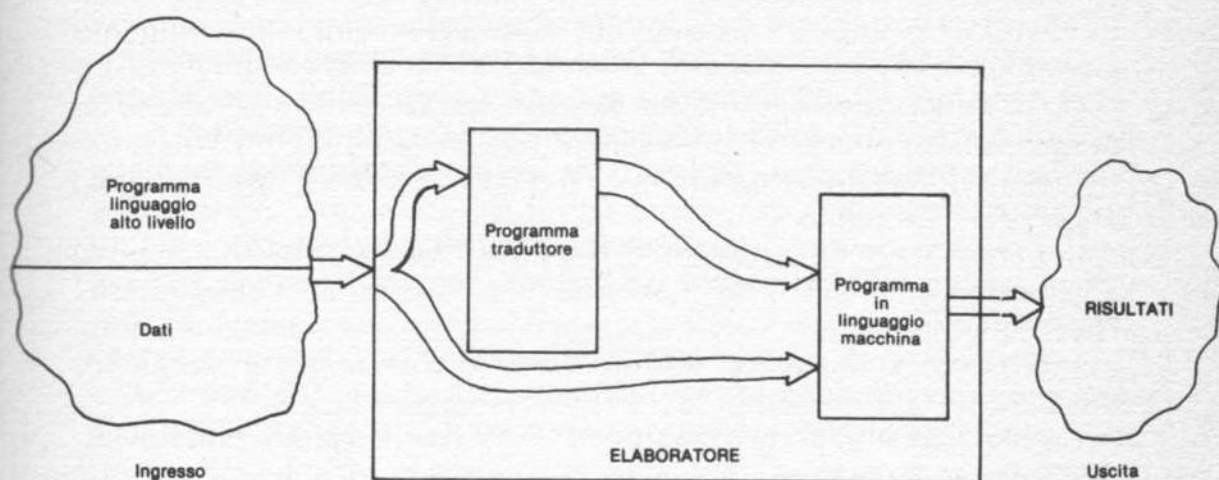
1. Scrivere, una volta per tutte, un programma traduttore che accetti, come dato in ingresso, il programma scritto in un linguaggio ad alto livello ed emetta, come risultato, la sua traduzione in linguaggio macchina.

Chiaramente tale traduzione sarà operativamente equivalente al programma originale, cioè, in parole povere, eseguirà gli stessi comandi scritti nel primo programma. Il programma traduttore è spesso chiamato «compilatore».

In parecchi home computer, comunque, il compilatore non esiste. E allora, si chiederà il Sig. Vit, come comunico con l'elaboratore? Niente paura. Esisterà un altro programma, più rudimentale, chiamato interprete, che invece di tradurre tutto il programma, una volta per tutte, tradurrà le sue singole istruzioni volta per volta.

Questo secondo programma è più lento e meno ottimizzato del primo. Comunque, per un home computer, va più che bene.

2. Ogni qualvolta si voglia, poi, fare eseguire un programma bisogna effettuare queste operazioni:

2a) traduzione, da parte del programma traduttore, del programma in linguaggio ad alto livello in programma in linguaggio macchina;

2b) esecuzione del programma in linguaggio macchina con dei dati opportuni.

Chiaramente la fase 2a) potrà essere effettuata, indifferentemente, da un compilatore o da un interprete. Riassumiamo in una sola illustrazione i passaggi logici per l'esecuzione di un programma.

Vi è ancora qualche osservazione da fare. In genere il programma traduttore segnalerà anche, come risultato dell'elaborazione, un elenco di errori sintattici commessi dal programmatore durante la stesura del programma.

Chiaramente non potrà rilevare eventuali errori logici, sui quali ci soffermeremo nel prossimo paragrafo, ma che comunque dovranno essere analizzati dal programmatore stesso.

A questo punto abbiamo capito che un elaboratore può parlare tramite un linguaggio ad alto livello.

Esaminiamo ora un po' più a fondo quello che si intende per linguaggio ad alto livello, le caratteristiche principali dei programmi, ed il modo con cui noi possiamo tradurre il nostro linguaggio naturale in linguaggio comprensibile all'elaboratore.

## 2.2 CARATTERISTICHE DEI PROGRAMMI

Dopo aver analizzato il particolare, nell'ambito della programmazione, saliamo un po' di livello per esaminare qualcosa di più generale.

Intendo dire, cioè, che dopo aver imparato le regole di un linguaggio di programmazione per la costruzione delle frasi (sintassi) e il loro significato (semantica), ciò che servirà al programmatore sarà l'acquisizione di criteri, in larga parte indipendenti dal linguaggio, per un buon progetto di programmi risolutivi di problemi.

Tali criteri potrebbero essere così catalogati:

— facilità nella costruzione del programma, intesa come naturalezza nella individuazione della soluzione, a partire dall'enunciazione del problema, e nella sua traduzione nel linguaggio prescelto;

— correttezza del programma, intesa come aderenza di esso allo specifico problema;

— trasparenza, intesa come semplicità nell'interpretazione di tutti i passi del programma stesso da parte di persone estranee alla sua costruzione;

— parametricità, intesa come la possibilità di un algoritmo di risolvere più problemi appartenti alla stessa classe del problema in esame, vale a dire problemi che possono essere interpretati allo stesso modo;

— modificabilità, intesa come semplicità dell'adattamento del programma a situazioni leggermente differenti da quella particolare in esame;

— efficienza, intesa come possibilità del programma di calcolare i dati in uscita a partire da quelli in ingresso, con il minimo spreco di risorse possibile;

— trasportabilità, intesa come possibilità di usare il programma su elaboratori diversi, senza la necessità di drastiche modifiche.

Tutti questi punti hanno un'importanza fondamentale nell'ambito della programmazione ed ogni buon programmatore dovrebbe conoscerli ed usarli nel corso del suo lavoro. Se analizziamo ciascun singolo punto, vediamo che ognuno ha una grossa importanza, sia come risparmio di mezzi hardware, sia come interscambiabilità, tra programmatori, di programmi che ciascuno ha costruito.

Passiamo ora ad altri due argomenti, spesso dimenticati, ma che sono molto importanti anch'essi, nel campo della programmazione: la documentazione e la correttezza dei programmi.

Per documentazione del programma si intende un qualcosa che aumenti la sua trasparenza, sia al progettista del programma stesso, sia a persone che non conoscono il programma ma che debbano lavorare con esso. Tutti i linguaggi di programmazione contengono delle particolari istruzioni, chiamate commenti, che non vengono eseguite dall'elaboratore, ma che servono esclusivamente a documentare il programma.

È comunque fondamentale che il programma stesso sia autodocumentante, ossia sia fatto in modo da poter ricostruire in breve tempo le operazioni che esegue. In ogni caso si potranno sempre inserire dei commenti per migliorarne la trasparenza.

I commenti inseribili si possono suddividere in due tipi:

— Commenti motivazionali, che esplicitano il significato del frammento di programma che essi precedono (ad esempio si può scrivere un commento che dice di azzerare una matrice: tutte le istruzioni, fino al prossimo commento, eseguiranno quel compito);

— Commenti asserzionali, che esprimono lo stato dei calcoli nel punto in cui vengono inseriti (ad esempio alla fine dell'operazione precedente di azzeramento della matrice, si scriverà un commento che in quel punto la matrice è azzerata).

Accanto a questo tipo di commenti ne esistono altri che dovrebbero sempre essere riportati in ogni programma. Per meglio dire ogni programma dovrebbe avere un'intestazione in cui viene riportata una breve descrizione dell'algoritmo ed il significato delle variabili più importanti usate durante l'esecuzione.

Può essere riportato un ulteriore tipo di commento. Cioè l'indentazione: disporre a livelli di colonna diversi dei frammenti di programma nidificati logicamente tra di loro. Ciò aumenterà di molto la leggibilità del vostro programma.

Per concludere questa breve parentesi sulla documentazione, vorrei esprimere un ultimo pensiero. Un programma ben documentato dovrebbe avere questa caratteristica: se si tolgono tutte le istruzioni eseguibili dall'elaboratore, i commenti letti in sequenza dovrebbero rivelare tutto lo svolgimento del programma, scritto, a questo punto, in linguaggio naturale.

Passiamo ora a vedere la correttezza dei programmi.

Abbiamo già visto cosa si intenda per correttezza di un programma. Si può vedere abbastanza facilmente che questa caratteristica del programma può essere divisa in più parti, o livelli. Vediamo quanti livelli di correttezza si possono individuare:

— Livello 1: il programma è sintatticamente corretto; ciò significa

che tale programma è accettabile dal compilatore che state usando, ma ciò non vi assicura, assolutamente, che risolva il vostro problema.

— Livello 2: ha acquisito il livello 1 e fornisce risposte corrette per «qualche» esempio di dati in ingresso per il vostro problema.

— Livello 3: ha acquisito il livello 2 e fornisce risposte corrette per un insieme di dati di test sufficientemente completo di tutti i casi possibili. È evidente che aumentando la complessità dell'algoritmo è sempre più complicato individuare insiemi di dati sufficientemente esaustivi.

— Livello 4: ha acquisito il livello 3 e fornisce una diagnostica di errore, emette cioè dei messaggi che rivelano l'errore, su insiemi di dati scorretti sufficientemente esaustivi.

— Livello 5: il programma fornisce risposta corretta a tutti i dati in ingresso, con diagnostica di errore per i dati in ingresso errati.

Dando un'occhiata a questi vari livelli, si nota una cosa fondamentale. I primi tre livelli di correttezza sono di solito quelli a cui si aspira, ma non sono assolutamente sufficienti per un buon programmatore. L'ultimo livello di correttezza non si potrà mai raggiungere. Rimane così il quarto livello che è quello a cui bisognerebbe in ogni caso aspirare.

## 2.3 DIAGRAMMI DI FLUSSO: COSA SONO?

Entriamo ora un poco più all'interno del significato della parola programmare. Abbiamo visto che ciò, inizialmente significa delegare ad un elaboratore la risoluzione di un determinato problema. Per ottenere questo scopo, bisogna trovare un algoritmo risolutivo del problema in esame, tradurre questo algoritmo in linguaggio comprensibile all'elaboratore ed infine fare eseguire alla macchina l'algoritmo prescelto.

La programmazione prescinde essenzialmente dal primo e dal terzo passo, occupandosi, nel contempo, della soluzione del secondo.

Per quanto riguarda tale problema, si può pensare di risolverlo in due ulteriori passi:

— un primo passo che traduca il nostro algoritmo in un tipo di linguaggio, che è chiamato linguaggio dei diagrammi di flusso;

— un secondo passo che codifichi l'algoritmo, nel linguaggio dell'elaboratore, partendo dalla descrizione dell'algoritmo in termini di diagrammi di flusso.

In questo paragrafo esamineremo il linguaggio dei diagrammi di flusso, dando così una prima occhiata al primo passo (il tutto sarà ripreso nel paragrafo degli esempi). Nel prossimo esamineremo il modo di codificare un algoritmo espresso in diagramma di flusso.

Notiano anzitutto che i tipi di istruzioni possibili, in un linguaggio di programmazione, sono essenzialmente questi:
— istruzioni di lettura o di input;
— istruzioni di scrittura o di output;
— istruzioni di calcolo e di assegnazione dei risultati del calcolo;
— istruzioni di confronto e ramificazione nell'esecuzione dell'algoritmo;
— istruzioni di trasferimento del controllo;
— istruzioni di inizio e fermata.

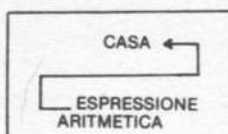
Chiameremo in seguito tali istruzioni con i nomi: lettura, scrittura, assegnazione, confronto e ramificazione, trasferimento del controllo, inizio e fermata.

Le istruzioni di confronto e ramificazione e di trasferimento del controllo sono anche dette istruzioni di controllo, perché controllano la sequenza di esecuzione delle altre istruzioni.

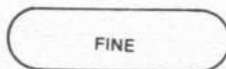
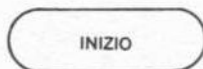
Vediamo ora i simboli usati nel linguaggio dei diagrammi di flusso, per tradurre queste istruzioni:

Quando si incontra questo simbolo bisogna leggere un dato ed assegnarlo alla variabile di nome FRAS. Passare poi alla istruzione successiva.

Quando si incontra uno di questi simboli bisogna scrivere il valore della variabile FRAS su stampante (a), quello della variabile FRES su

video (b), quello della variabile FRIS su nastro. Di solito le istruzioni che traducono i simboli (a) e (b) sono le stesse: nel caso della stampante, bisognerà poi dare qualche altro comando per specificare all'elaboratore che vogliamo l'output su questo strumento (*):

Quando si incontra questo simbolo, bisogna calcolare il valore dell'espressione e poi assegnare il risultato alla variabile di nome CASA.

Se si incontra questo simbolo bisogna, inizialmente, calcolare il valore dell'espressione logica: poi se la risposta è VERO percorrere il ramo di controllo sinistro, se è FALSO seguire quello destro.

Se si incontra il simbolo suddetto non bisogna far altro che seguire il senso della freccia.

Questi due simboli indicano soltanto l'inizio e la fine dell'algoritmo.

Tutti questi simboli corrispondono ai vari tipi di istruzione che abbiamo in precedenza elencato. Perciò in pochi simboli siamo riusciti a raccogliere tutte le istruzioni possibili, contenute in un linguaggio di programmazione.

A questo punto il Sig. Vip si potrebbe chiedere: perché si sono introdotti i diagrammi di flusso?

(*) Nel seguito i nostri diagrammi di flusso useranno il simbolo di input, sia per simulare istruzioni di lettura che quelle di scrittura.

Ci sono almeno tre valide risposte.

La prima è che abbiamo introdotto i diagrammi di flusso per mostrare un tipo di linguaggio, diverso dal naturale, in cui si possano esprimere gli algoritmi. Da ciò si deduce anche che gli algoritmi possono essere comunicati con qualsiasi tipo di linguaggio: l'importante è che il committente e l'esecutore dell'algoritmo siano perfettamente d'accordo sui significati dei simboli.

La seconda risposta è che in questo modo si è analizzato un linguaggio in cui compaiono le prime differenze nella rappresentazione dei vari tipi di istruzioni.

La terza, e forse più importante, è che i diagrammi di flusso sono il passo intermedio tra la descrizione di un algoritmo in linguaggio naturale e la sua codifica in linguaggio di programmazione.

È anche il passo intermedio più adatto, perché una volta scritto l'algoritmo nel linguaggio dei diagrammi di flusso, il suo passaggio alla codifica è banale: si tratta soltanto di una traduzione dei simboli che abbiamo appena visto in istruzioni di programma.

Una cosa, forse decisiva a favore dei diagrammi di flusso è che, una volta scritto l'algoritmo in questo linguaggio, si potrà tradurlo con qualsiasi linguaggio di programmazione e per qualsiasi elaboratore.

Infatti fino a questo momento non abbiamo specificato nulla dell'hardware e del software dell'elaboratore usato. Soltanto nella fase di traduzione di questo schema, nel linguaggio prescelto, si inseriranno le caratteristiche peculiari di ogni sistema elaborativo.

## 2.4 USO DEI DIAGRAMMI DI FLUSSO

Vediamo ora il secondo passo del processo di programmazione. Vale a dire la codifica, nel linguaggio di programmazione scelto (*), dell'algoritmo, espresso nel linguaggio dei diagrammi di flusso.

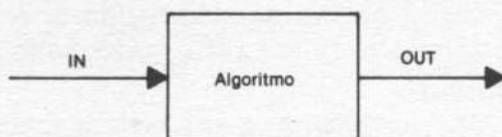
Tale passo è il più banale. Perché? Perché, se il diagramma di flusso è costruito con regole precise, la sua codifica non sarà altro che una traduzione. Vale a dire ad ogni simbolo del diagramma di flusso corrisponderà una, ed una sola, istruzione (o un gruppo di istruzioni).

Perciò non dovremo fare altro che andare a vedere sul nostro «vocabolario», che mette in corrispondenza i simboli dei diagrammi di flusso con le istruzioni, il simbolo che stiamo cercando di codificare.

(*) In questo testo useremo soltanto il BASIC.

A questo punto rivediamo in modo globale il processo di programmazione.

Di solito ogni algoritmo ha un gruppo di ingresso (o un'ingresso formato da più dati elementari), ed un gruppo di uscita (o un'uscita formata da più dati elementari).

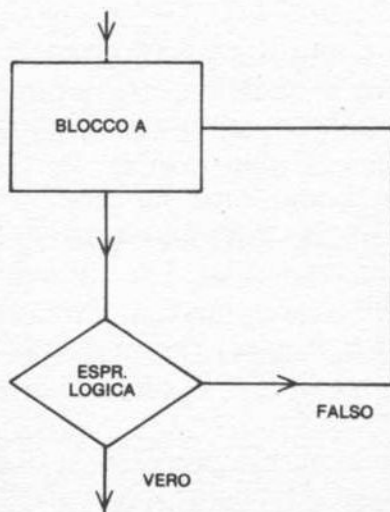
Perciò un algoritmo si può rappresentare in questo modo

dove con IN ed OUT si possono intendere anche insiemi di dati.

È fondamentale, nel corso del processo della programmazione, mantenere la struttura ad un ingresso e un'uscita.

Mi spiego meglio.

Le istruzioni di controllo potrebbero avere un ingresso e più uscite: ma ciò diminuirebbe di molto la leggibilità e la trasparenza del programma.

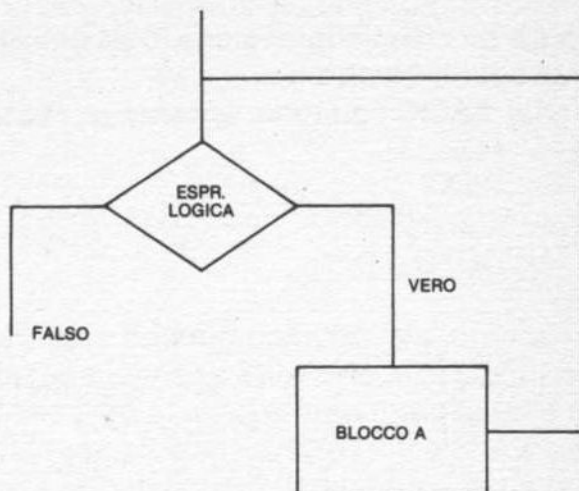
In più avendo soltanto strutture ad un ingresso e un'uscita, si potrà procedere nel programmare tramite raffinamenti successivi. Infatti partendo dal blocco ALGORITMO della figura precedente si espanderà tale blocco in più blocchetti, che avranno però sempre un ingresso e un'uscita. Per espansioni successive si arriverà al livello di dettaglio utile per essere codificato.

Le strutture di controllo disponibili per attuare ciò che abbiamo appena detto sono le seguenti (espresse con il linguaggio dei diagrammi di flusso):

con la codifica

LOOP

  blocco A

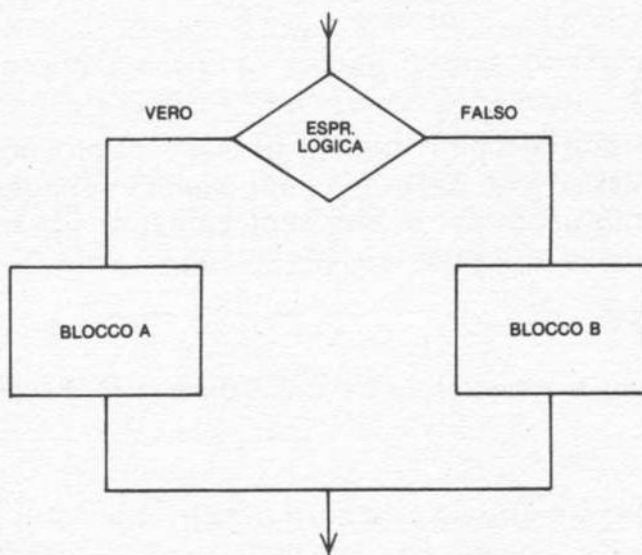
UNTIL espr. logica REPEAT

vale a dire che ripete il blocco A fino a che l'espressione logica non è verificata essere vera;

con la codifica

WHILE espr. logica DO

  blocco A

REPEAT

vale a dire: se l'espressione logica è vera esegui il blocco A, altrimenti esci;

con codifica

IF espr. logica THEN

  blocco A

ELSE

blocco B

ENDIF

vale a dire: se l'espressione logica è vera esegui il blocco A, altrimenti esegui il blocco B.

Nel BASIC esistono soltanto le seguenti strutture di controllo:

— FOR...TO
  NEXT
— IF...THEN
— GOTO

Con dei piccoli accorgimenti si potranno ricostruire le strutture ad un ingresso e un'uscita appena elencate, che potranno essere usate in fase di programmazione. In ogni caso si può notare che le strutture di controllo del BASIC, se usate separatamente tra di loro, sono ad un ingresso e a un'uscita (escludendo l'IF...THEN).

Non penso ci sia altro da aggiungere a questo semplice passo, ma forse sarà utile seguire lo sviluppo di due semplici programmi dalla loro nascita alla loro codifica, mettendone in risalto ulteriori caratteristiche di una buona programmazione.

## 2.5 ESEMPI

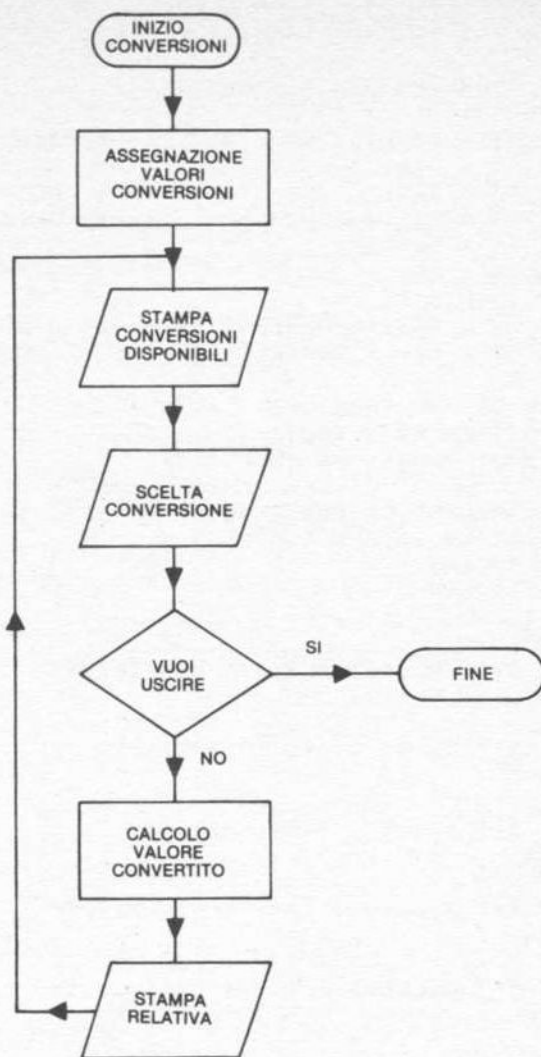
Vediamo ora due semplici esempi di come si procede nella programmazione. Esporremo dapprima il problema in linguaggio naturale, poi l'algoritmo usato per la sua risoluzione, la sua traduzione in diagrammi di flusso ed infine la sua codifica (*).

### PROBLEMA 1

Calcolare alcune conversioni tra unità di misura diverse.

*Algoritmo usato*

Si è scelto di memorizzare i rapporti di conversione in un vettore e poi, in base ad una richiesta da parte dell'utilizzatore, scegliere il rapporto a noi necessario e calcolare così il valore convertito.

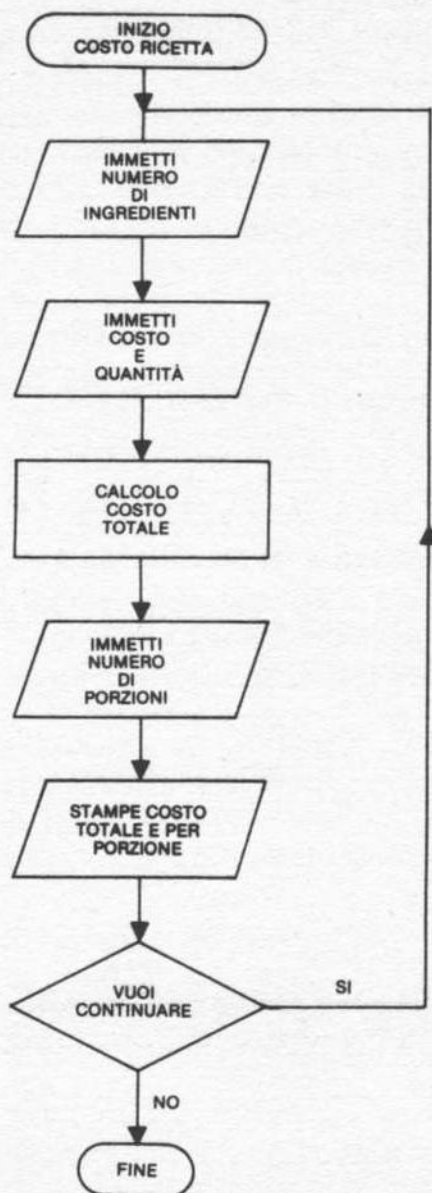
(*) Questo sarà anche il modo con cui nei prossimi paragrafi presenteremo gli altri programmi.

```
10    REM
20    REM CONVERSIONI UNITA' DI MISURA
30    REM
40    REM  VARIABILI USATE:
50    REM  C(8) PER CONTENERE I RAPPORTI DI CONVERSIONE
60    REM
70    DIM C(8)
80    REM ASSEGNAZIONI RAPPORTI DI CONVERSIONE
90    FOR I=1 TO 8
100     READ C(I)
110   NEXT I
120   REM TABELLA RAPPORTI DI CONVERSIONE
130   DATA 2.540,.3048,.9144,1.609
```

```
140  DATA 3.785,28.3495,.4536,1
150  CLS
160  PRINT "CONVERSIONI DISPONIBILI:"
170  PRINT
180  PRINT "1 = POLLICI              ---*  CENTIMETRI"
190  PRINT "2 = PIEDI                ---*  METRI"
200  PRINT "3 = YARDE                ---*  METRI"
210  PRINT "4 = MIGLIA               ---*  CHILOMETRI"
220  PRINT "5 = GALLONI              ---*  LITRI"
230  PRINT "6 = ONCE                 ---*  GRAMMI"
240  PRINT "7 = LIBBRE               ---*  CHILOGRAMMI"
250  PRINT "8 = GRADI FAHRENHEIT  ---*  GRADI CELSIUS"
260  PRINT "SCEGLI LA CONVERSIONE CHE TI OCCORE (O PER USCIRE)"
270  INPUT N
280  IF N<0 OR N>8 THEN GOTO 270
290  IF N=0 THEN GOTO 9000
300  REM INPUT VALORE DA CONVERTIRE
310  CLS
320  INPUT "VALORE DA CONVERTIRE";A
330  REM CALCOLO VALORE CONVERTITO
340  LET B=A*C(N)
350  GOSUB 1000*N
360  GOTO 150
1000 PRINT
1010 PRINT A;" POLLICI = ";B;" CENTIMETRI"
1020 RETURN
2000 PRINT
2010 PRINT A;" PIEDI = ";B;" METRI"
2020 RETURN
3000 PRINT
3010 PRINT A;" YARDE = ";B;" METRI"
3020 RETURN
4000 PRINT
4010 PRINT A;" MIGLIA = ";B;" CHILOMETRI"
4020 RETURN
5000 PRINT
5010 PRINT A;" GALLONI = ";B;" LITRI"
5020 RETURN
6000 PRINT
6010 PRINT A;" ONCE = ";B;" GRAMMI"
6020 RETURN
7000 PRINT
7010 PRINT A;" LIBBRE = ";B;" CHILOGRAMMI"
7020 RETURN
8000 REM CONVERSIONE GRADI
8010 LET B=(A-32)*5/9
8020 PRINT A;" GRADI FAHRENHEIT = ";B;" GRADI CELSIUS"
8030 RETURN
9000 END
```

## PROBLEMA 2

Calcolare il costo di una ricetta complessivo e per porzione.

*Algoritmo usato*

Bisognerà immettere il costo e la quantità necessaria di ogni ingrediente per la ricetta. L'elaboratore poi calcolerà il suo costo totale per porzione.

```
10    REM
20    REM COSTO DI UNA RICETTA
30    REM
40    REM VARIABILI USATE:
50    REM N NUMERO INGREDIENTI
60    REM C(N) COSTO UNITARIO
70    REM F(N) UNITA' DI MISURA RICHIESTE
80    REM
90    CLS
100   INPUT 'QUANTI INGREDIENTI CI SONO';N
110   DIM C(N),F(N)
120   FOR I=1 TO N
130     PRINT 'INGREDIENTE ';I
140     PRINT 'IMMETTI IL COSTO PER UNITA' DI MISURA'
150     INPUT C(I)
160     PRINT 'IMMETTI QUANTITA' NECESSARIA'
170     INPUT F(I)
180   NEXT I
190   REM CALCOLO COSTO
200   TOT=0
210   FOR J=1 TO N
220     LET A=C(J)*F(J)
230     LET TOT=TOT+A
240   NEXT J
250   CLS
260   PRINT 'QUANTE PORZIONI FAI?'
270   INPUT P
280   LET PPOR=INT(TOT/P)
290   REM STAMPE
300   CLS
310   PRINT
320   PRINT 'LA RICETTA TI COSTERA': ';TOT;' LIRE'
330   PRINT
340   PRINT 'PER PORZIONE TI COSTERA': ';PPOR;' LIRE'
350   CLS
360   PRINT 'VUOI CALCOLARE IL COSTO DI UN'ALTRA RICETTA (1=SI,0=NO)?'
370   INPUT S
380   IF S=1 THEN GOTO 90
390   END
```

# 3. PROGRAMMI DI INTERESSE PER LA FAMIGLIA

Nel presente capitolo e nei successivi tre verranno illustrati, seguendo il loro progetto e costruzione, diversi programmi. In essi potrete verificare come le tecniche che sono state sinora descritte vengano sempre applicate per costruire dei buoni programmi.

L'unico consiglio che si può dare, prima di entrare nel mondo dei programmi, è quello di non copiarli subito per provarli, ma di cercare innanzitutto di capire la loro struttura per poi poterli usare in maniera più efficace.

In tal modo potrete adattarli ai vostri scopi particolari, che chiaramente un libro scritto da un'altra persona non potrà mai raggiungere.

## LA DIETA (I)

Quante volte nella vita vi è capitato di fare una dieta? Penso almeno una. Quanta difficoltà avrete incontrato nello scegliere i vari pasti per mantenere il giusto numero di calorie? Penso molte.

Ecco una cosa per cui vi può essere utile il vostro elaboratore casalingo.

Voi dovrete solamente dichiarare il numero di calorie che vi è necessario e l'elaboratore vi dirà cosa dovrete mangiare nella giornata.

In più ve lo dirà in modo tale che risparmierete anche sul costo degli ingredienti: vale a dire mangerete il numero di calorie giusto al minimo costo.

Si sono costruiti due programmi per questo scopo: il DIETA 1 e il DIETA 2.

Mentre nel primo sono memorizzati gli ingredienti, che perciò sono fissi, nel secondo voi potrete scegliere anche gli ingredienti fino ad un numero di 6.

Descriviamo ora il DIETA 1.

Abbiamo qui a disposizione 9 ingredienti: pasta o pane, carne, pe-

sce, legumi, verdura o frutta, olio, latte, formaggio, uova.

Voi dovrete inserire le calorie necessarie nella giornata insieme con lo scarto utile; vale a dire se vi servono 2000 calorie potrebbero anche andarvi bene 1900 o 2100 calorie. Infine inserirete i 9 costi degli ingredienti.

Il programma, con un algoritmo comprendente una serie di FOR, prova tutte le combinazioni e alla fine emetterà i risultati.

Se ha trovato qualcosa vi dirà la lista degli ingredienti, con il loro peso, a prezzo ottimo. Se non ha trovato niente vi emetterà un messaggio spiegandone il motivo.

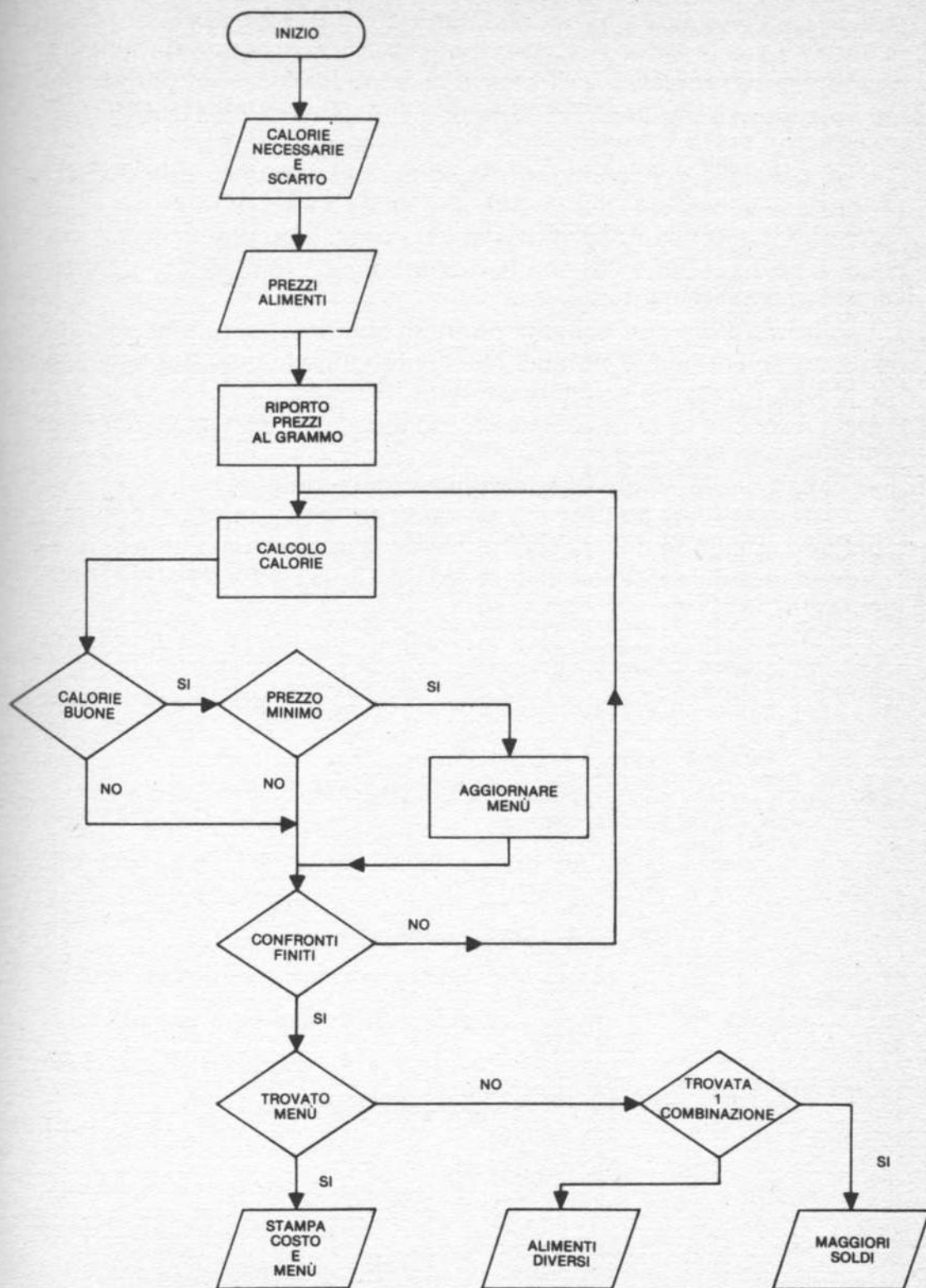
Le strutture dei dati consistono in un vettore che contiene i costi degli ingredienti e in 9 variabili che contengono la quantità necessaria di essi. Un'ultima variabile conterrà il costo ottimo.

Da notare che se la risposta sarà «ho bisogno di diversi alimenti» si potrà agire in due modi:

— allargare lo scarto di accettabilità delle calorie;

— cambiare nel programma un qualche alimento, dove citato, e cambiare le calorie del vecchio alimento con quelle del nuovo.

Vediamo ora dapprima il diagramma di flusso e poi il listato in linguaggio BASIC.

INIZIO
CALORIE
NECESSARIE
E
SCARTO
PREZZI
ALIMENTI
RIPORTO
PREZZI
AL GRAMMO
CALCOLO
CALORIE
CALORIE
BUONE
SI
PREZZO
MINIMO
SI
NO
NO
AGGIORNARE
MENÙ
CONFRONTI
FINITI
NO
SI
TROVATO
MENÙ
NO
TROVATA
1
COMBINAZIONE
SI
SI
STAMPA
COSTO
E
MENÙ
ALIMENTI
DIVERSI
MAGGIORI
SOLDI

```
10   REM DIETA GIORNALIERA
15   REM
20   REM CALCOLO DEL CIBO CHE SODDISFA ALLA DIETA AL MINIMO PREZZO
30   REM
40   REM VARIABILI USATE
50   REM P = PASTA O PANE
60   REM C = CARNE
70   REM PE = PESCE
80   REM L = LEGUMI
90   REM V = VERDURA O FRUTTA
100 REM O = OLIO
110 REM LA = LATTE
120 REM F = FORMAGGI
130 REM U = UOVA
140 REM
150 CLS
155 DIM CO(9)
160 LET MO=100000
170 LET Z=0
180 INPUT 'QUANTE CALORIE VUOI';CN
185 INPUT 'CON QUALE SCARTO';S
190 INPUT 'COSTO DI 100 GRAMMI DI PANE O PASTA';CO(1)
200 INPUT 'COSTO DI 100 GRAMMI DI CARNE';CO(2)
210 INPUT 'COSTO DI 100 GRAMMI DI PESCE';CO(3)
220 INPUT 'COSTO DI 100 GRAMMI DI LEGUMI';CO(4)
230 INPUT 'COSTO DI 100 GRAMMI DI VERDURA O FRUTTA';CO(5)
240 INPUT 'COSTO DI 100 GRAMMI DI OLIO';CO(6)
250 INPUT 'COSTO DI 100 GRAMMI DI LATTE';CO(7)
260 INPUT 'COSTO DI 100 GRAMMI DI FORMAGGI';CO(8)
270 INPUT 'COSTO DI UN UOVO';CO(9)
280 FOR I=1 TO 8
290   LET CO(I)=CO(I)/100
300 NEXT I
310 REM   CALCOLO DIETA
320 FOR P=50 TO 150 STEP 25
330   FOR C=50 TO 200 STEP 50
340     FOR PE=50 TO 200 STEP 50
350       FOR L=50 TO 150 STEP 25
360         FOR V=50 TO 200 STEP 50
370           FOR O=2 TO 6 STEP 2
380             FOR LA=100 TO 300 STEP 100
390               FOR F=100 TO 200 STEP 50
400                 FOR U=1 TO 3
410                   LET CC=P*3.5+C*2.0+P*1.0+L*3.0+V*0.4+O*9+LA*
0.5+F*3+U*400
420                   IF CC<CN-S OR CC>CN+S THEN GOTO 560
430                   LET Z=Z+1
440                   LET PO=P*CO(1)+C*CO(2)+PE*CO(3)+L*CO(4)+V*CO
(5)+O*CO(6)+ LA*CO(7)+F*CO(8)+U*CO(9)
450                   IF PO>MO THEN GOTO 560
460                   LET MO=PO
470                   LET P1=P
480                   LET C1=C
490                   LET PE1=PE
500                   LET L1=L
510                   LET V1=V
520                   LET O1=O
```

```
530                     LET LA1=LA
540                     LET F1=F
550                     LET U1=U
555 REM
560 REM FINE CALCOLO DI UNA POSSIBILE DIETA OTTIMA
565 REM
570                   NEXT U
580                 NEXT F
590               NEXT LA
600             NEXT O
610           NEXT V
620         NEXT L
630       NEXT PE
640     NEXT C
650 NEXT P
655 REM
660 REM STAMPE
665 REM
670 CLS
680 IF MO>=100000 THEN GOTO 820
690 PRINT 'SCELTE POSSIBILI:';Z
700 PRINT 'LA SCELTA A COSTO MINIMO:';MO,'E':'
710 PRINT 'DEVI COMPRARE:'
720 PRINT P1,'GRAMMI DI PANE O PASTA'
730 PRINT C1,'GRAMMI DI CARNE'
740 PRINT PE1,'GRAMMI DI PESCE'
750 PRINT L1,'GRAMMI DI LEGUMI'
760 PRINT V1,'GRAMMI DI VERDURA O FRUTTA'
770 PRINT O1,'GRAMMI DI OLIO'
780 PRINT LA1,'GRAMMI DI LATTE'
790 PRINT F1,'GRAMMI DI FORMAGGIO'
800 PRINT U1,'UOVA'
805 GOTO 870
810 PRINT
815 REM
820 REM CON QUESTI DATI NON E' POSSIBILE MANGIARE
825 REM
830 IF Z>0 THEN GOTO 860
840 PRINT 'HO BISOGNO DI DIVERSI ALIMENTI'
850 GOTO 870
860 PRINT 'HO BISOGNO DI PIU' SOLDI'
870 END
```

## LA DIETA (2)

Come abbiamo detto, questo programma ricalca un poco il precedente con la differenza che sarete voi a suggerire gli ingredienti per la dieta.

Le strutture di dati questa volta sono 3 vettori di 6 elementi contenenti:

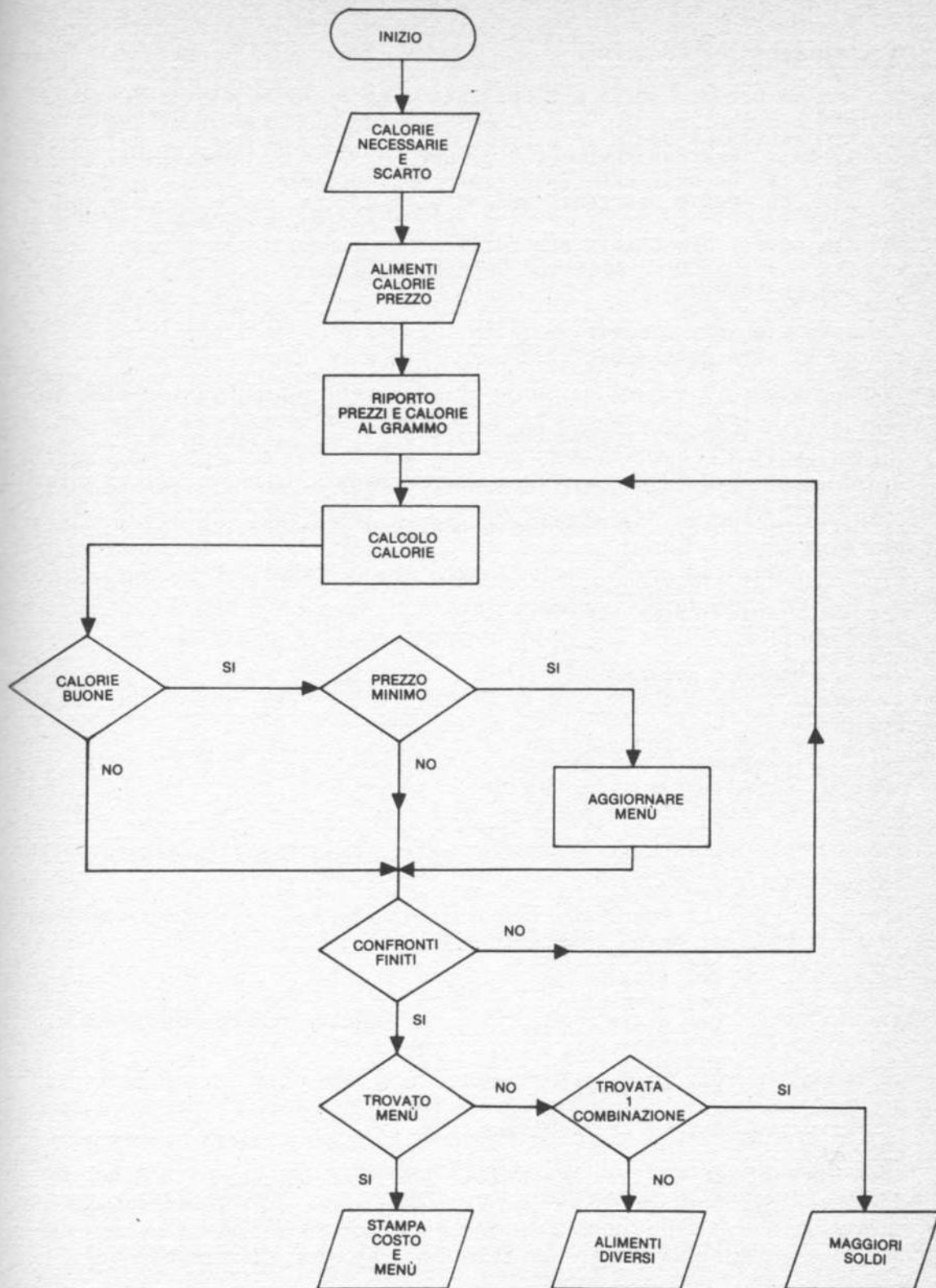
— il nome dell'ingrediente;
— il suo costo;
— le sue calorie.

Da notare che potete scegliere 6 ingredienti. Questo per motivi di tempo. Infatti aumentando il numero degli elementi aumenta il numero dei confronti, in modo esponenziale, perciò anche l'aumento di due o tre ingredienti potrebbero far salire il tempo macchina in modo da non poter vedere i risultati per tutta la giornata. Nell'attesa potreste anche morire di fame!

Comunque per diminuire il numero di confronti si potrebbe agire sui FOR cambiandoli da come sono ora, da 50 a 200 grammi passo 25, ad esempio da 100 a 200 grammi passo 50. Così si passerebbe, per ogni FOR, da 7 confronti a 3 (*).

Vediamo ora il diagramma di flusso e poi il listato relativo in BASIC.

(*) Il tempo macchina, per i FOR nidificati, è proporzionale al numero di confronti del FOR elevato al numero dei FOR presenti (il tempo macchina è il tempo di occupazione della CPU, che è in stretto rapporto con il tempo che trascorrerà prima che voi vediate la risposta). Perciò il numero di confronti effettuati, per 6 ingredienti passerà da $7^6 = 117.649$ a $3^6 = 729$!

INIZIO
CALORIE NECESSARIE E SCARTO
ALIMENTI CALORIE PREZZO
RIPORTO PREZZI E CALORIE AL GRAMMO
CALCOLO CALORIE
CALORIE BUONE
SI
PREZZO MINIMO
SI
NO
NO
AGGIORNARE MENÙ
CONFRONTI FINITI
NO
SI
TROVATO MENÙ
NO
TROVATA 1 COMBINAZIONE
SI
SI
NO
STAMPA COSTO E MENÙ
ALIMENTI DIVERSI
MAGGIORI SOLDI

```
10  REM DIETA GIORNALIERA
15  REM
20  REM CALCOLO DEL CIBO CHE SODDISFI ALLA DIETA AL MINIMO PREZZO
30  REM
40  REM VARIABILI USATE
50  REM  A$  PER CONTENERE LE PIETANZE
60  REM  CA  CALORIE PER CENTO GRAMMI DI PIETANZA
70  REM  CO  COSTO PER CENTO GRAMMI DI PIETANZA
80  REM
85  DIM A$(6): DIM CA(6): DIM CO(6)
90  CLS
100 LET MO=100000
110 LET Z=0
120 INPUT "QUANTE CALORIE VUOI";CN
125 INPUT "CON QUALE SCARTO";S
130 PRINT
140 FOR I=1 TO 6
145   CLS
150   INPUT "QUALE ALIMENTO VUOI MANGIARE (MAX 6)";A$(I)
160   PRINT
170   INPUT "QUANTE CALORIE CI SONO OGNI CENTO GRAMMI";CA(I)
180   PRINT
190   INPUT "QUAL E' IL SUO COSTO OGNI CENTO GRAMMI";CO(I)
200 NEXT I
202 FOR K=1 TO 6
204   LET CA(K)=CA(K)/100
206   LET CO(K)=CO(K)/100
208 NEXT K
210 REM
215 REM CALCOLO DIETA
220 REM
230 FOR B=50 TO 200 STEP 25
240   FOR C=50 TO 200 STEP 25
250     FOR D=50 TO 200 STEP 25
260       FOR E=50 TO 200 STEP 25
270         FOR F=50 TO 200 STEP 25
280           FOR G=50 TO 200 STEP 25
290           LET CC=B*CA(1)+C*CA(2)+D*CA(3)+E*CA(4)+F*CA(5)+G*CA(6)
300           IF CC<CN-S OR CC>CN+S THEN GOTO 410
310           LET Z=Z+1
320           LET PO=B*CO(1)+C*CO(2)+D*CO(3)+E*CO(4)+F*CO(5)+G*CO(6)
330           IF PO>MO THEN GOTO 410
340           LET MO=PO
350           LET B1=B
360           LET C1=C
370           LET D1=D
380           LET E1=E
390           LET F1=F
400           LET G1=G
410 REM
415 REM FINE CALCOLO DIETA OTTIMA
420 REM
430           NEXT G
440         NEXT F
```

```
450        NEXT E
460      NEXT D
470    NEXT C
480 NEXT B
485 REM
490 REM STAMPE
495 REM
500 CLS
505 IF MO>=100000 THEN GOTO 620
510 PRINT "SCELTE POSSIBILI:";Z
520 PRINT "LA SCELTA A COSTO MINIMO E':";MO;"LIRE"
530 PRINT
540 PRINT "COMPRA:"
550 PRINT B1;"GRAMMI DI",A$(1)
560 PRINT C1;"GRAMMI DI",A$(2)
570 PRINT D1;"GRAMMI DI",A$(3)
580 PRINT E1;"GRAMMI DI",A$(4)
590 PRINT F1;"GRAMMI DI",A$(5)
600 PRINT G1;"GRAMMI DI",A$(6)
610 GOTO 670
620 REM NON E' POSSIBILE MANGIARE
630 IF Z>0 THEN GOTO 660
640 PRINT "HO BISOGNO DI DIVERSI ALIMENTI"
650 GOTO 670
660 PRINT "HO BISOGNO DI PIU' SOLDI!"
670 END
```

---

## GESTIONE FAMILIARE

Lo scopo del programma in esame è di leggere, memorizzare e riemettere, sotto forma di output facilmente leggibile, tutte le spese e le entrate del mese corrente. Il programma calcolerà anche i totali del mese corrente e terrà in memoria quelli dei mesi precedenti, ma dell'anno in corso.

Il programma sarà strutturato con un menù, vale a dire con diverse opzioni disponibili di volta in volta. Quest'ultimo sarà così composto.

1. Aggiornamento spese ed entrate del giorno. Avremo la possibilità, per quanto riguarda la prima opzione di:

- — aggiornare le entrate (disposte su una sola colonna);
- — aggiornare le uscite che saranno così suddivise:
  - — vitto,
  - — casa,
  - — servizi,
  - — trasporti,
  - — abbigliamento,
  - — varie.

Se vorremmo cambiare queste intestazioni non dovremmo fare altro che variare le istruzioni di PRINT tra la 1060 e la 1230 e quelle di output delle matrici nelle due altre subroutine di stampa.

Bisogna comunque rimanere nel numero di 7 campi tra entrate ed uscite; se si vorrà aumentare il numero si dovranno ridimensionare tutte le matrici ed i vettori, facendo però un poco di conti con la memoria disponibile.

2. Stampa matrice del mese in corso, in cui vengono stampati tutti i dati relativi al mese in corso, in righe successive rappresentanti i vari giorni. L'ultima riga rappresenterà i totali.

3. Stampa totali mesi precedenti, in cui vengono stampati i dati relativi ai mesi dell'anno in corso. Al solito l'ultima riga rappresenta i totali.

4. Scarico totali mese precedente in una matrice riepilogativa, che dovrà essere effettuata alla fine di ogni mese.

5. Salvataggio dei dati da effettuarsi alla fine di ogni sessione di lavoro.

Diamo qualche cenno sulle strutture di dati usate.

Sono state utilizzate due matrici A (32 × 8) e B (13 × 8). Nella prima sono memorizzati i dati relativi al mese in corso, mentre nella seconda quelli relativi ai mesi precedenti e al mese in corso (sotto forma di totali).

Da notare che l'ultima riga e l'ultima colonna di ogni matrice sono dei totali, la prima per quanto riguarda i vari campi, la seconda per quel che riguarda i vari giorni o mesi.

Vediamo ora la prima traduzione in diagramma di flusso e quindi la codifica definitiva in linguaggio BASIC.

INIZIO
GESTIONE
INPUT
MESE
ATTUALE
OPZIONE
AGGIORNAMENTI
GIORNO
STAMPA
MESE
ATTUALE
STAMPA
TOTALI
MESI
SCARICO
MESE
PRECEDENTE
SALVATAGGIO
DATI
INIZIO
REGISTRAZIONE
VUOI
REGISTRARE?
W$
REGISTRAZIONE
DATI
W$ = M
RETURN
INIZIO
STAMPA MESE
STAMPA
MATRICE
MESE IN
CORSO
RITORNO
AL
MENÙ
INIZIO
STAMPA RIEPILOGO
STAMPA
MATRICE
DEI MESI
RITORNO
AL
MENÙ

INIZIO
AGGIORNAMENTO
IMMISSIONE
GIORNO
IMMISSIONE
CAMPO
CAMPO = O
SI
RETURN
NO
IMMISSIONE
QUANTITÀ
AGGIORNAMENTI
DUE MATRICI

INIZIO
SCARICO
VUOI
SCARICARE?
NO
RETURN
SI
SCARICO E
AGGIORNAMENTO
MATRICE
MESE ATTUALE

```
10    REM
20    REM  PROGRAMMA GESTIONE SPESE
30    REM
40    REM  VARIABILI USATE:
50    REM  A(32,8)  PER IL MESE CORRENTE
60    REM  B(13,8)  PER IL RIEPILOGO DEI MESI PRECEDENTI
70    REM
80    DIM A(32,8): DIM B(13,8)
90    CLS
140   REM RICHIESTA DEL MESE ATTUALE
150   PRINT
155   PRINT
160   INPUT "QUAL'E' IL MESE ATTUALE";MR
170   IF MR<0 OR MR>12 THEN GOTO 160
180   REM
190   REM INIZIO MENU'
200   REM
210   CLS
220   PRINT "===========>  MENU'"
230   PRINT
240   PRINT "1 - AGGIORNAMENTO GIORNO"
250   PRINT
260   PRINT "2 - STAMPA MESE ATTUALE "
270   PRINT
280   PRINT "3 - STAMPA TOTALI MESI  "
290   PRINT
300   PRINT "4 - SCARICO MESE PRECEDENTE"
310   PRINT
320   PRINT "5 - SALVATAGGIO DATI"
330   PRINT
340   PRINT "SCEGLI (0 = FINE LAVORO)"
350   INPUT N
360   IF N<0 OR N>5 THEN GOTO 350
370   IF INT(N)<>N THEN GOTO 350
380   IF N=0 THEN GOTO 410
390   GOSUB N*1000
400   GOTO 90
410   STOP
1000 REM
1005 REM AGGIORNAMENTO GIORNO
1010 REM
1015 CLS
1020 INPUT "QUALE GIORNO VUOI AGGIORNARE";G
1030 IF G<0 OR G>31 THEN GOTO 1020
1035 REM
1040 REM ABBIAMO IL GIORNO CORRETTO DA AGGIORNARE
1045 REM
1050 CLS
1060 PRINT "SCELTA DEL CAMPO DA AGGIORNARE:"
1070 PRINT
1080 PRINT "1 = ENTRATE"
1090 PRINT
1100 PRINT "2 = VITTO"
1110 PRINT
1120 PRINT "3 = CASA"
1130 PRINT
1140 PRINT "4 = SERVIZI"
```

```
1150 PRINT
1160 PRINT "5 = TRASPORTI"
1170 PRINT
1180 PRINT "6 = ABBIGLIAMENTO"
1190 PRINT
1200 PRINT "7 = VARIE"
1210 PRINT
1220 PRINT "0 = USCITA"
1230 PRINT
1240 PRINT "SCEGLI:"
1250 INPUT A
1260 IF A<0 OR A>7 THEN GOTO 1035
1270 IF A=0 THEN GOTO 1480
1280 REM A= AGGIORNAMENTO GIUSTO
1290 CLS
1300 PRINT
1310 INPUT "QUANTITA' DA IMMETTERE";IM
1320 IF INT(IM)<>IM THEN GOTO 1310
1330 LET A(G,A)=A(G,A)+IM
1340 REM AGGIORNAMENTO TOTALI MATRICE
1350 LET TOT=0
1360 FOR I=2 TO 7
1370   LET TOT=TOT+A(G,I)
1380 NEXT I
1390 LET A(G,8)=TOT
1400 FOR I=1 TO 8
1410   LET TOTT=0
1420   FOR J=1 TO 31
1430     LET TOTT=TOTT+A(J,I)
1440   NEXT J
1450   LET A(32,I)=TOTT
1460 NEXT I
1470 GOTO 1035
1480 REM
1490 REM AGGIORNAMENTO TABELLA MESI
1500 REM
1510 FOR I=1 TO 8
1520   LET B(MR,I)=A(32,I)
1530 NEXT I
1540 FOR I=1 TO 8
1550   LET TOTA=0
1560   FOR J=1 TO 12
1570     LET TOTA=TOTA+B(J,I)
1580   NEXT J
1590   LET B(13,I)=TOTA
1600 NEXT I
1610 RETURN
2000 REM ROUTINE DI STAMPA
2005 CLS
2010 PRINT "MESE DI ";MR;" I GIORNI SONO CONSECUTIVI, L'ULTIMO DAT
O E' IL TOTALE"
2020 PRINT " ENTRATE ,  VITTO  , CASA  , SERVIZI ,TRASPORTI, ABBIG
L. , VARIE  ,  TOTUSC "
2030 PRINT
2040 FOR I=1 TO 32
2050 PRINT A(I,1),A(I,2),A(I,3),A(I,4),A(I,5),A(I,6),A(I,7),A(I,8)
```

```
2060 NEXT I
2070 RETURN
3000 REM ROUTINE PER STAMPA TOTALI MESI
3010 CLS
3020 PRINT "CONSUNTIVO ANNO IN CORSO ; I MESI SONO CONSECUTIVI"
3030 PRINT "L'ULTIMA RIGA EFFETTUA I TOTALI"
3040 PRINT " ENTRATE ,  VITTO  ,  CASA  , SERVIZI ,TRASPORTI, ABBI
GL. ,VARIE  ,  TOTUSC "
3050 PRINT
3060 FOR I=1 TO 13
3070   PRINT       B(I,1),B(I,2),B(I,3),B(I,4),B(I,5),B(I,6),B(I,7
),B(I,8)
3080 NEXT I
3090 RETURN
4000 REM SCARICO MESE IN CORSO : DA EFFETTUARE OGNI FINE DEL MESE!
4010 INPUT "VUOI SALVARE IL MESE IN CORSO (S=SI,N=NO)";W$
4020 IF W$="S" THEN GOTO 4040
4030 RETURN
4040 FOR I=1 TO 8
4050   LET B(MR,I)=A(32,I)
4060 NEXT I
4070 FOR I=1 TO 8
4080   LET TOTAL=0
4090   FOR J=1 TO 12
4100     LET TOTAL=TOTAL+B(J,I)
4110   NEXT J
4120   LET B(13,I)=TOTAL
4130 NEXT I
4140 FOR I=1 TO 8
4150   FOR J=1 TO 32
4160     LET A(I,J)=0
4170   NEXT J
4180 NEXT I
4190 RETURN
5000 REM R E G I S T R A Z I O N E   D A T I
5010 REM
5020 PRINT "REGISTRAZIONE DATI"
5030 PRINT "PREMI <RETURN> , <M>=TORNA AL MENU'"
5040 INPUT W$
5050 IF W$="M" THEN RETURN
5060 CLS
5070 PRINT "RIAVVOLGI IL NASTRO"
5080 PRINT "QUANDO SEI PRONTO PER REGISTRARE PREMI"
5090 PRINT "<RETURN>"
5100 INPUT W$
5120 SAVE "GESTIONE SPESE" LINE 5130
5130 GOTO 90
```

## AGENDA TELEFONICA

Vediamo ora l'agenda telefonica. Questo ormai è un programma classico tra gli home computer ed è forse tra i più utili.

La lista del programma conterrà sei opzioni:

— opzione di lista, per avere un elenco completo del contenuto di tutta l'agenda;

— opzione di inserimento, per inserire nuovi nominativi. Per problemi di spazio i nominativi dovranno essere, al massimo, lunghi:

- — 15 caratteri il cognome,
- — 10 caratteri il nome,
- — 20 caratteri l'indirizzo,
- — 5 caratteri il C.A.P.,
- — 10 caratteri la città,
- — 12 caratteri il numero telefonico.

Se si dispone di spazio di memoria RAM, cioè quella utilizzabile direttamente per i programmi e i dati relativi ad essi, maggiormente di 16 Kbyte si potranno allargare, di conseguenza, questi campi agendo sui dimensionamenti dei vettori fatti ad inizio programma dalla riga 60 alla riga 110;

— opzione di ricerca, per ricercare i nominativi in base a più chiavi.

Le chiavi sono:

- — cognome e nome,
- — cognome,
- — nome,
- — indirizzo,
- — telefono;

— opzione di cancellazione, per effettuare la cancellazione di indirizzi ormai diventati vecchi;

— opzione di azzeramento, per azzerare tutte le variabili in gioco nel programma, e di conseguenza tutta l'agenda;

— opzione di salvataggio, che dovrà essere attivata per memorizzare tutti i nuovi indirizzi e numeri telefonici inseriti volta per volta.

Le strutture di dati usate sono essenzialmente queste:

C $: matrice che contiene i cognomi,
N $: matrice che contiene i nomi,
I $: matrice che contiene gli indirizzi,
A $: matrice che contiene i C.A.P.,
T $: matrice che contiene le città,
P $: matrice che contiene i numeri di telefono,
A: contatore dei record memorizzati.

Tutti i vettori hanno 135 posizioni, sempre per problemi di spazio. Se si ha un elaboratore con memoria più vasta si potrà allungare la loro dimensione.

Facendo ciò, comunque, bisognerà tenere presente che questo tipo di algoritmo è stato studiato per piccole dimensioni, vale a dire fino a 700-800 record. Se si ha bisogno di più record bisognerà studiare algoritmi diversi che ottimizzino il tempo di ricerca; altrimenti si rischia di aspettare per ore una risposta dall'elaboratore!

Diamo ora i diagrammi di flusso relativi, traducibili come al solito in qualsiasi linguaggio, e una loro codifica in linguaggio BASIC.

INIZIO
LISTA
INPUT
W$
W$ = M
SI
RETURN
NO
CONTATORE
K = 1
K>A
SI
RETURN
NO
LISTA
W$ = M
SI
RETURN
NO
K = K + 1

INIZIO
INSERIMENTO
INCREMENTO
CONTATORE
A = A + 1
A > 135
SI
MEMORIA
OCCUPATA
NO
IMMETTI
W$
W$ = M
SI
NO
IMMETTI
I
CAMPI
DEL
RECORD
SI
VUOI
CORREGGERE?
NO
VUOI
ANNULLARE?
SI
DECREMENTO
CONTATORE
A = A—1
RETURN
NO
RETURN

INIZIO RICERCA
CONTATORE C = 0 VUOI CERCARE?
IMMETTI W$
W$ = M
SI
RETURN
NO
INPUT OPZIONE RICERCA
OPZIONE RICERCA
INPUT COGNOME E NOME
INPUT COGNOME
INPUT NOME
INPUT INDIRIZZO
INPUT NUMERO TELEFONO
PORTATI LUNG GIUSTA RICERCA NOME
PORTATO LUNG GIUSTA RICERCA NOME
PORTATO LUNG GIUSTA RICERCA NOME
PORTATO LUN GIUSTA RICERCA NOME
PORTATO LUNG GIUSTA RICERCA NOME
CONTATORE K>A
TROVATO
SI
NO
GOSUB 1
GOSUB 1
③
②
①

INIZIO
CANCELLAZIONE
VUOI
CANCELLARE?
W$ = M
SI
RETURN
NO
NOME
DA
CANCELLARE
SUBROUTINE
RICERCA
CONTATORE
>A
SI
RETURN
NO
CONTATORE
= A
SI
NO
SPOSTA ULTIMO
RECORD SU
QUELLO PUNTATO
DA CONTATORE
CANCELLA
ULTIMO
RECORD
RETURN
1
VISUALIZZAZIONE
STAMPA
RECORD
INPUT
W$
CONTATORE C
INCREMENTATO
RETURN
2
CONTATORE>0
SI
RETURN
NO
3
NOMINATIVO
NON IN
MEMORIA
RETURN

```
10   REM PROGRAMMA GESTORE DI UN ELENCO TELEFONICO
20   REM
30   REM DIMENSIONAMENTO DELLE VARIABILI
40   REM COGNOME, NOME, INDIRIZZO, C.A.P., CITTA', TELEFONO
50   REM
60   DIM C$(135,15)
70   DIM N$(135,10)
80   DIM I$(135,20)
90   DIM A$(135,5)
100  DIM T$(135,10)
110  DIM P$(135,12)
120  LET A=0
130  LET L$="COGNOME"
140  LET D$="NOME"
150  LET O$="INDIRIZZO"
160  LET P$="CAP"
165  LET S$="CITTA'"
170  LET Q$="TELEFONO"
180  CLS
190  PRINT " 1 - LISTA "
200  PRINT " 2 - INSERIMENTO "
210  PRINT " 3 - RICERCA "
220  PRINT " 4 - CANCELLAZIONE "
230  PRINT " 5 - AZZERAMENTO "
240  PRINT " 6 - SALVATAGGIO "
```

```
250  INPUT 'SCEGLI';Z
260  IF Z<1 OR Z>6 THEN GOTO 180
270  CLS
280  GOSUB Z*1000
290  GOTO 180
1000 REM L I S T A
1010 REM
1020 PRINT 'LISTA RUBRICA'
1030 PRINT 'PREMI <RETURN> , <M> = MENU''
1040 INPUT W$
1050 IF W$='M' THEN RETURN
1060 FOR K=1 TO A
1070   GOSUB 3600
1080   IF W$='M' THEN RETURN
1090 NEXT K
1100 RETURN
2000 REM I N S E R I M E N T O
2010 REM
2020 LET A=A+1
2030 IF A<135 THEN GOTO 2060
2040 PRINT 'NON POSSO INSERIRE: MEMORIA PIENA!'
2050 GOTO 2370
2060 PRINT 'INSERIMENTO DATI'
2070 PRINT 'PREMI <RETURN> , <M> = MENU''
2080 INPUT W$
2090 IF W$='M' THEN GOTO 2370
2100 CLS
2110 PRINT 'INSERISCI I DATI COME SEGUE'
2120 PRINT L$
2130 PRINT D$
2140 PRINT O$
2150 PRINT P$
2155 PRINT S$
2160 PRINT Q$
2170 INPUT C$(A)
2180 PRINT C$(A)
2190 INPUT N$(A)
2200 PRINT N$(A)
2210 INPUT I$(A)
2220 PRINT I$(A)
2230 INPUT A$(A)
2240 PRINT A$(A)
2250 INPUT T$(A)
2260 PRINT T$(A)
2270 INPUT P$(A)
2280 PRINT P$(A)
2290 PRINT 'VUOI CORREGGERE? (S/N)'
2300 INPUT W$
2310 IF W$<>'S' THEN GOTO 2340
2320 CLS
2330 GOTO 2110
2340 PRINT 'VUOI ANNULLARE? <S/N>'
2350 INPUT W$
2360 IF W$='N' OR W$='' THEN RETURN
2370 LET A=A-1
2380 RETURN
3000 REM R I C E R C A
```

```
3010 REM
3020 LET C=0
3030 PRINT 'RICERCA'
3040 PRINT 'PREMI <RETURN> , <M>=MENU''
3045 INPUT W$
3050 IF W$='M' THEN RETURN
3055 CLS
3060 PRINT 'RICERCA CON:'
3062 PRINT '1 - COGNOME E NOME'
3064 PRINT '2 - COGNOME'
3066 PRINT '3 - NOME'
3068 PRINT '4 - INDIRIZZO'
3070 PRINT '5 - TELEFONO'
3072 PRINT 'SCEGLI?'
3075 INPUT Z
3080 IF Z<1 OR Z>5 THEN GOTO 3075
3085 CLS
3090 GOTO Z*100+3000
3100 PRINT 'INSERISCI IL COGNOME'
3105 INPUT B$
3110 IF LEN (B$)>15 THEN GOTO 3105
3115 IF LEN (B$)=15 THEN GOTO 3130
3120 LET B$=B$+' '
3125 GOTO 3115
3130 PRINT 'INSERISCI IL NOME'
3135 INPUT E$
3140 IF LEN (E$)>10 THEN GOTO 3135
3145 IF LEN (E$)=10 THEN GOTO 3160
3150 LET E$=E$+' '
3155 GOTO 3145
3160 FOR K=1 TO A
3165   IF C$(K)=B$ AND N$(K)=E$ THEN GOTO 3600
3170 NEXT K
3175 GOTO 3660
3200 PRINT 'INSERISCI IL COGNOME'
3205 INPUT B$
3210 IF LEN (B$)>15 THEN GOTO 3205
3215 IF LEN (B$)=15 THEN GOTO 3230
3220 LET B$=B$+' '
3225 GOTO 3215
3230 FOR K=1 TO A
3240   IF C$(K)=B$ THEN GOSUB 3600
3250 NEXT K
3260 GOTO 3650
3300 PRINT 'INSERISCI IL NOME'
3305 INPUT B$
3310 IF LEN (B$)>10 THEN GOTO 3305
3315 IF LEN (B$)=10 THEN GOTO 3330
3320 LET B$=B$+' '
3325 GOTO 3315
3330 FOR K=1 TO A
3340   IF N$(K)=B$ THEN GOSUB 3600
3350 NEXT K
3360 GOTO 3650
3400 PRINT 'INSERISCI L'INDIRIZZO'
3405 INPUT B$
3410 IF LEN (B$)>20 THEN GOTO 3405
```

```
3415 IF LEN (B$)=20 THEN GOTO 3430
3420 LET B$=B$+' '
3425 GOTO 3415
3430 FOR K=1 TO A
3440   IF I$(K)=B$ THEN GOTO 3600
3450 NEXT K
3460 GOTO 3660
3500 PRINT 'INSERISCI IL NUMERO DI TELEFONO'
3505 INPUT B$
3510 IF LEN (B$)>12 THEN GOTO 3505
3515 IF LEN (B$)=12 THEN GOTO 3530
3520 LET B$=B$+' '
3525 GOTO 3515
3530 FOR K=1 TO A
3540   IF P$(K)=B$ THEN GOTO 3600
3550 NEXT K
3560 GOTO 3660
3600 CLS
3602 PRINT L$;' ';C$(K)
3604 PRINT D$;' ';N$(K)
3606 PRINT O$;' ';I$(K)
3608 PRINT P$;' ';A$(K),S$;' ';T$(K)
3610 PRINT Q$;' ';P$(K)
3615 PRINT 'PER CONTINUARE PREMI <RETURN>'
3620 INPUT W$
3630 LET C=C+1
3640 RETURN
3650 IF C>0 THEN RETURN
3660 CLS
3665 LET K=K+1
3670 PRINT 'RECORD NON PRESENTE'
3680 RETURN
4000 REM C A N C E L L A Z I O N E
4010 REM
4020 PRINT 'CANCELLAZIONE RECORD'
4030 PRINT 'PREMI <RETURN> PER CONTINUARE; <M>=MENU''
4040 INPUT W$
4050 IF W$='M' THEN RETURN
4060 CLS
4070 PRINT 'QUALE NOMINATIVO VUOI CANCELLARE?'
4080 GOSUB 3100
4090 IF K>A THEN RETURN
4100 IF K=A GOTO 4170
4110 LET C$(K)=C$(A)
4120 LET N$(K)=N$(A)
4130 LET I$(K)=I$(A)
4140 LET A$(K)=A$(A)
4150 LET T$(K)=T$(A)
4160 LET P$(K)=P$(A)
4170 LET C$(A)=''
4180 LET N$(A)=''
4190 LET I$(A)=''
4200 LET A$(A)=''
4210 LET T$(A)=''
4220 LET P$(A)=''
4230 LET A=A-1
4240 CLS
```

```
4250 PRINT 'NOMINATIVO CANCELLATO'
4260 RETURN
5000 REM A Z Z E R A M E N T O
5010 REM
5020 PRINT 'VUOI VERAMENTE AZZERARE? (S/N)'
5030 INPUT W$
5040 IF W$<>'S' AND W$<>N THEN GOTO 5030
5050 IF W$='N' THEN RETURN
5060 CLS
5070 FOR K=1 TO A
5080   LET C$(K)=''
5090   LET N$(K)=''
5100   LET I$(K)=''
5110   LET A$(K)=''
5120   LET T$(K)=''
5130   LET P$(K)=''
5140 NEXT K
5150 LET A=0
5160 RETURN
6000 REM R E G I S T R A Z I O N E   D A T I
6010 REM
6020 PRINT 'REGISTRAZIONE DATI'
6030 PRINT 'PREMI <RETURN> , <M>=MENU''
6040 INPUT W$
6050 IF W$='M' THEN RETURN
6060 CLS
6070 PRINT 'RIAVVOLGI IL NASTRO'
6080 PRINT 'QUANDO SEI PRONTO PER REGISTRARE PREMI'
6090 PRINT '<RETURN>'
6100 INPUT W$
6120 SAVE 'AGENDA' LINE 6130
6130 GOTO 180
```

## ANNAFFIATURA PIANTE

Quante volte vi sarete chiesti: «Da quanto tempo non dò l'acqua a questa pianta? Mi sembra che abbia la terra un po' secca»!
Ebbene il vostro elaboratore vi potrà aiutare anche in questo compito.
Infatti con questo programma, giorno per giorno, potrete sapere quali piante, del vostro giardino o balcone, dovrete annaffiare. Il programma è strutturato, ancora una volta, a lista. Le opzioni sono:
— inizializzazione, in cui dovrete dichiarare le piante che possedete, il loro periodo di annaffiatura, il giorno ed il mese dell'ultima annaffiatura. A questa dovrete anche ricorrere quando avrete una nuova pianta (*). Il numero di piante che si possono memorizzare sono 20; comunque se avete una memoria più potente potrete, al solito, memorizzarne di più agendo sui DIM di inizio programma, dalla riga 40 alla 45;
— aggiornamento annaffiatura, cui dovrete ricorrere per memorizzare ogni annaffiatura effettuata;
— suggerimenti per annaffiatura odierna, che vi dirà se avreste già dovuto annaffiare delle piante durante i giorni precedenti, se nel giorno in corso ne dovete annaffiare qualcuna o se invece non avete nulla da annaffiare;
— salvataggio dati, che dovrà essere attivato ogni volta che terminate una sessione con il vostro elaboratore.
Le strutture di dati usate sono essenzialmente tre vettori ed una matrice:

| | |
|---|---|
| P$ (20 × 20) | contenente il nome delle 20 piante memorizzate; |
| P (20) | contenente il periodo di annaffiatura relativo alle piante precedenti; |
| GU (20) e MU (20) | contenente il giorno ed il mese dell'ultima annaffiatura della pianta relativa. |

Diamo di seguito il diagramma di flusso dell'algoritmo ed una sua traduzione in BASIC.

(*) O quando dovrete aggiornare il periodo di annaffiatura.

INIZIO INIZIALIZZAZIONE

IMMETTI:
• PIANTA
• PERIODO
• ULTIMA ANNAFFIATURA

SIAMO A 20

NO

SI

RETURN

INIZIO AGGIORNAMENTO

INPUT PIANTA ANNAFFIATA

RICERCA DELLA PIANTA

ESISTE

SI

NO

AGGIORN. GIORNO E MESE

LA PIANTA NON ESISTE

RETURN

INIZIO
ANNAFFIATURA
IMMETTI
DATA
CONTATORE
=0
MESE =
ULTIMA
ANNAFF.
CALCOLO
GIORNI
DEL MESE
PASSATO = V
<P
V
≥P
CALCOLO
GIORNI
MESE
PRESENTE = V
AVRESTI
GIÀ
DOVUTO
ANNAFFIARE
<P
V
>P
=P
DEVI
ANNAFFIARE
OGGI
NO
PIANTE
FINITE
SI
CONTATORE
=0
NON DEVI
ANNAFFIARE
<> 0
RETURN
INIZIO
SALVATAGGIO
VUOI
SALVARE?
W$ = M
SI
RETURN
NO
REGISTRAZIONE
DATI

```
10   REM INNAFFIATURA PIANTE
20   REM
25   REM VARIABILI USATE:
27   REM  P$(20,20)  PER CONTENERE I NOMI DELLE PIANTE
29   REM  P(20) PER CONTENERE IL PERIODO DI INNAFFIATURA DELLE PIA
NTE
31   REM  GU(20) ED MU(20) PER CONTENERE GIORNO E MESE DELL'ULTIMA
33   REM                      INNAFFIATURA
35   REM
40   DIM P$(20,20)
45   DIM P(20): DIM GU(20): DIM MU(20): DIM GM(12)
47   LET GM(1)=31
48   LET GM(2)=28
49   LET GM(3)=31
50   LET GM(4)=30
51   LET GM(5)=31
52   LET GM(6)=30
53   LET GM(7)=31
54   LET GM(8)=31
55   LET GM(9)=30
56   LET GM(10)=31
57   LET GM(11)=30
58   LET GM(12)=31
59   REM
60   CLS
65   PRINT
70   PRINT "===========>  M E N U'"
75   PRINT
80   PRINT "1 - INIZIALIZZAZIONE"
85   PRINT
90   PRINT "2 - AGGIORNAMENTO ANNAFFIATURA"
95   PRINT
100  PRINT "3 - ANNAFFIATURA ODIERNA"
105  PRINT
110  PRINT "4 - SALVATAGGIO DEI DATI"
120  INPUT "QUALE SCEGLI";Z
130  IF Z<1 OR Z>4 THEN GOTO 120
140  GOSUB Z*1000
150  GOTO 60
1000 REM INIZIALIZZAZIONE
1010 CLS
1020 FOR I=1 TO 20
1025   PRINT "PIANTA ATTUALE: ";P$(I)
1030   PRINT "IMMETTI PIANTA NUOVA:"
1040   INPUT "PIANTA (<RETURN> PER CONFERMARE)";A$
1050   IF A$="" THEN GOTO 1070
1060   LET P$(I)=A$
1070   PRINT "PERIODO ANNAFFIATURA ATTUALE: ";P(I)
1075   INPUT "PERIODO ANNAFFIATURA (<0> PER CONFERMARE)";A
1080   IF A=0 THEN GOTO 1100
1090   LET P(I)=A
1100   PRINT "GIORNO ULTIMA ANNAFFIATURA: ";GU(I)
1110   INPUT "GIORNO ULTIMA ANNAFFIATURA (<0> PER CONFERMARE)";B
1120   IF B=0 THEN GOTO 1130
1125   LET GU(I)=B
1130   PRINT "MESE ULTIMA ANNAFFIATURA: ";MU(I)
1135   INPUT "MESE ULTIMA ANNAFFIATURA (<0> PER CONFERMARE)";C
```

```
1140   IF C=0 THEN GOTO 1160
1150   LET MU(I)=C
1160 NEXT I
1170 RETURN
2000 REM AGGIORNAMENTO ANNAFFIATURA
2010 CLS
2020 PRINT "QUALE PIANTA HAI ANNAFFIATO?"
2030 INPUT B$
2040 IF LEN(B$)>20 THEN GOTO 2030
2050 IF LEN (B$)=20 THEN GOTO 2080
2060 LET B$=B$+" "
2070 GOTO 2050
2080 FOR I=1 TO 20
2090   IF P$(I)=B$ THEN GOTO 2130
2100 NEXT I
2110 PRINT
2115 PRINT "PIANTA NON ESISTENTE!!!!!!!!"
2120 RETURN
2130 REM AGGIORENAMENTO DATI
2140 PRINT "AGGIORNAMENTO GIORNO:"
2150 INPUT B
2160 LET GU(I)=B
2165 PRINT
2170 PRINT "AGGIORNAMENTO MESE"
2180 INPUT C
2190 LET MU(I)=C
2200 RETURN
3000 REM ANNAFFIATURA ODIERNA
3010 CLS
3020 PRINT "IMMETTI LA DATA ODIERNA"
3025 PRINT
3030 INPUT "GIORNO";G
3033 IF G<0 OR G>31 THEN GOTO 3030
3035 PRINT
3040 INPUT "MESE";M
3043 IF M<0 OR M>12 THEN GOTO 3040
3045 LET W=0
3050 FOR I=1 TO 20
3055   LET V=0
3060   IF M=MU(I)THEN GOTO 3100
3070   LET V=GM(M)-GU(I)
3080   IF V>P(I) THEN GOTO 3085
3082   LET V=V+G
3083   IF V<P(I) THEN GOTO 3160
3084   IF V=P(I) THEN GOTO 3125
3085   LET W=W+1
3086   IF V=P(I) THEN GOTO 3130
3090   PRINT "AVRESTI GIA' DOVUTO ANNAFFIARE LA PIANTA: ";P$(I)
3095   GOTO 3160
3100   LET V=V+(G-GU(I))
3110   IF V<P(I) THEN GOTO 3160
3120   IF V>P(I) THEN GOTO 3145
3125   LET W=W+1
3127   PRINT
3130   PRINT "OGGI DEVI ANNAFFIARE: ";P$(I)
3140   GOTO 3160
3145   LET W=W+1
```

```
3147   PRINT
3150   PRINT "AVRESTI GIA' DOVUTO ANNAFFIARE: ";P$(I)
3160 NEXT I
3170 IF W<>0 THEN RETURN
3180 PRINT
3185 PRINT "NON DEVI ANNAFFIARE NESSUNA PIANTA!!!"
3190 RETURN
4000 REM R E G I S T R A Z I O N E   D A T I
4010 REM
4020 PRINT "REGISTRAZIONE DATI"
4030 PRINT "PREMI <RETURN> , <M>=MENU'"
4040 INPUT W$
4050 IF W$="M" THEN RETURN
4060 CLS
4070 PRINT "RIAVVOLGI IL NASTRO"
4080 PRINT "QUANDO SEI PRONTO PER REGISTRARE PREMI"
4090 PRINT "<RETURN>"
4100 INPUT W$
4120 SAVE "PIANTE" LINE 4130
4130 GOTO 60
```

## MENÙ OFFERTI AGLI OSPITI

Vi è mai capitato di invitare un amico più volte e di sentirvi dire: «Buona questa pietanza, l'avevo già mangiata da te»!

Il seguente programma serve proprio per evitare tali situazioni poco piacevoli. Infatti memorizza gli ospiti ed insieme ad essi il menù che gli è stato preparato, di volta in volta.

Le opzioni del menù sono:

— opzione di lista, per vedere tutto il contenuto dell'elenco. Tale lista, una volta che è stata raggiunta la saturazione di spazio di memoria, può essere effettuata su stampante prima dell'azzeramento di tutte le variabili;

— opzione di inserimento, per inserire nuovi menù. Lo spazio a disposizione è il seguente:

- — 15 caratteri per il cognome,
- — 10 caratteri per i primi piatti,
- — 20 caratteri per i secondi piatti,
- — 15 caratteri per i contorni,
- — 15 caratteri per il dessert;

— opzione di ricerca, fatto sulle chiavi:

- — cognome,
- — primo,
- — secondi,
- — contorni,
- — dessert;

— opzione di azzeramento, per cancellare tutte le variabili, una volta raggiunta la situazione di memoria;

— opzione di salvataggio, da attivare ogni volta che inseriamo un nuovo menù.

Le strutture di dati usate sono le seguenti:

O$: matrice contenente i cognomi degli ospiti;
P$: matrice contenente i primi piatti;
S$: matrice contenente i secondi piatti;
C$: matrice contenente i contorni;
D$: matrice contenente i dessert;
A: contatore dei record memorizzati.

Tutti i vettori hanno 135 posizioni, che come al solito, avendo una memoria RAM più grande di 16 Kbyte, potranno essere espansi. Vediamo ora il diagramma di flusso generale, relativo all'algoritmo sopra esposto, e poi una sua traduzione in BASIC.

INIZIO
AZZERAMENTO
VUOI
AZZERARE?
W$ = M
SI
RETURN
NO
AZZERAMENTO
DATI
INIZIO
SALVATAGGIO
VUOI
SALVARE?
W$ = M
SI
RETURN
NO
PREMI
RETURN
QUANDO SEI
PRONTO
SALVA
PROGRAMMA
DATI
INIZIO
LISTA
INPUT
W$
W$ = M
SI
RETURN
NO
CONTATORE
K = 1
K>A
SI
RETURN
NO
LISTA
W$ = M
SI
RETURN
NO
INCREMENTO
CONTATORE
K = K + 1

INIZIO
INSERIMENTO
INCREMENTO
CONTATORE
A = A + 1
A > 135
SI
MEMORIA
OCCUPATA
NO
IMMETTI
W$
W$ = M
SI
NO
IMMETTI I
CAMPI
DEL
RECORD
SI
VUOI
CORREGGERE?
NO
VUOI
ANNULLARE
SI
DECREMENTO
CONTATORE
A = A—1
RETURN
NO
RETURN

INIZIO
RICERCA
CONTATORE
C=0
VUOI
CERCARE?
IMMETTI
W$
W$=M
SI
RETURN
NO
INPUT
COGNOME
PORTATO
LUNGHEZZ
GIUSTA
RICERCA
NOME
SI
2
CONTATORE
K>A
NO
TROVATO
NO
SI
GOSUB
1
1
VISUALIZZAZIONE
STAMPA
RECORD
INPUT
W$
CONTATORE C
INCREMENTATO
RETURN
2
CONTATORE>0
SI
RETURN
NO
NOMINATIVO
NON IN
MEMORIA
RETURN

```
10    REM PROGRAMMA GESTORE DI MENU' OFFERTI AGLI OSPITI
20    REM
30    REM DIMENSIONAMENTO DELLE VARIABILI
40    REM COGNOME, PRIMI, SECONDI, CONTORNI, DESSERT
50    REM
60    DIM O$(135,15)
70    DIM P$(135,10)
80    DIM S$(135,20)
100   DIM C$(135,15)
110   DIM D$(135,15)
120   LET A=0
130   LET L$='GLI OSPITI'
140   LET A$='PRIMI'
150   LET B$='SECONDI'
160   LET E$='CONTORNI'
170   LET F$='DESSERT'
180   CLS
190   PRINT ' 1 - LISTA '
200   PRINT ' 2 - INSERIMENTO '
210   PRINT ' 3 - RICERCA '
230   PRINT ' 4 - AZZERAMENTO '
240   PRINT ' 5 - SALVATAGGIO '
250   INPUT 'SCEGLI';Z
260   IF Z<1 OR Z>5 THEN GOTO 180
270   CLS
280   GOSUB Z*1000
290   GOTO 180
1000 REM L I S T A
1010 REM
1020 PRINT 'LISTA RUBRICA'
1030 PRINT 'PREMI <RETURN> , <M> = MENU''
1040 INPUT W$
1050 IF W$='M' THEN RETURN
1060 FOR K=1 TO A
1070   GOSUB 3600
1080   IF W$='M' THEN RETURN
1090 NEXT K
1100 RETURN
2000 REM I N S E R I M E N T O
2010 REM
2020 LET A=A+1
2030 IF A<135 THEN GOTO 2060
2040 PRINT 'NON POSSO INSERIRE: MEMORIA PIENA!'
2050 GOTO 2370
2060 PRINT 'INSERIMENTO DATI'
2070 PRINT 'PREMI <RETURN> , <M> = MENU''
2080 INPUT W$
2090 IF W$='M' THEN GOTO 2370
2100 CLS
2110 PRINT 'INSERISCI I DATI COME SEGUE'
2115 PRINT L$
2120 PRINT A$
2130 PRINT B$
2140 PRINT E$
2150 PRINT F$
2170 INPUT O$(A)
2180 PRINT O$(A)
```

```
2190 INPUT P$(A)
2200 PRINT P$(A)
2210 INPUT S$(A)
2220 PRINT S$(A)
2250 INPUT C$(A)
2260 PRINT C$(A)
2270 INPUT D$(A)
2280 PRINT D$(A)
2290 PRINT 'VUOI CORREGGERE? (S/N)'
2300 INPUT W$
2310 IF W$<>'S' THEN GOTO 2340
2320 CLS
2330 GOTO 2110
2340 PRINT 'VUOI ANNULLARE? <S/N>'
2350 INPUT W$
2360 IF W$='N' OR W$='' THEN RETURN
2370 LET A=A-1
2380 RETURN
3000 REM R I C E R C A
3010 REM
3020 LET C=0
3030 PRINT 'RICERCA'
3040 PRINT 'PREMI <RETURN> , <M>=MENU''
3045 INPUT W$
3050 IF W$='M' THEN RETURN
3100 PRINT 'INSERISCI IL COGNOME'
3105 INPUT B$
3110 IF LEN (B$)>15 THEN GOTO 3105
3115 IF LEN (B$)=15 THEN GOTO 3160
3120 LET B$=B$+' '
3125 GOTO 3115
3160 FOR K=1 TO A
3165   IF O$(K)=B$ THEN GOSUB 3600
3170 NEXT K
3175 GOTO 3650
3600 CLS
3602 PRINT L$;' ';O$(K),'HANNO MANGIATO:'
3604 PRINT A$;' ';P$(K)
3606 PRINT B$;' ';S$(K)
3608 PRINT E$;' ';C$(K)
3610 PRINT F$;' ';D$(K)
3615 PRINT 'PER CONTINUARE PREMI <RETURN>'
3620 INPUT W$
3630 LET C=C+1
3640 RETURN
3650 IF C>0 THEN RETURN
3660 CLS
3670 PRINT 'OSPITI MAI VENUTI'
3680 RETURN
4000 REM A Z Z E R A M E N T O
4010 REM
4020 PRINT 'VUOI VERAMENTE AZZERARE? (S/N)'
4030 INPUT W$
4040 IF W$<>'S' AND W$<>N THEN GOTO 4030
4050 IF W$='N' THEN RETURN
4060 CLS
```

```
4070 FOR K=1 TO A
4080   LET O$(K)=""
4090   LET P$(K)=""
4100   LET S$(K)=""
4120   LET C$(K)=""
4130   LET D$(K)=""
4140 NEXT K
4150 LET A=0
4160 RETURN
5000 REM R E G I S T R A Z I O N E   D A T I
5010 REM
5020 PRINT "REGISTRAZIONE DATI"
5030 PRINT "PREMI <RETURN> , <M>=MENU'"
5040 INPUT W$
5050 IF W$="M" THEN RETURN
5060 CLS
5070 PRINT "RIAVVOLGI IL NASTRO"
5080 PRINT "QUANDO SEI PRONTO PER REGISTRARE PREMI"
5090 PRINT "<RETURN>"
5100 INPUT W$
5120 SAVE "MENU'" LINE 5130
5130 GOTO 180
```

---

## CONSUMO AUTOMOBILE

Questo programma vi sarà utile per conoscere il consumo della vostra auto, ma richiede da parte vostra, una particolare attenzione e precisione, poiché, chiaramente, dovrete inserire i dati riguardanti i chilometri percorsi e la benzina comprata.

Dovrete anche, una volta per tutte inizializzare le variabili F e C; esse rappresentano:

**F:** i chilometri che la vostra automobile ha percorso dal momento in cui iniziate il controllo del consumo;

**C:** il costo della benzina nello stesso momento.

Il programma è composto da una lista; le sue opzioni sono:

— inserimento dati, in cui dovrete dichiarare:

1) il costo della benzina, se è variato,
2) il mese sul quale volete agire,
3) la quantità di benzina comprata in lire o in litri,
4) i chilometri percorsi;

— calcolo e stampa delle medie, che vi visualizzerà mensilmente il

consumo espresso in chilometri percorsi con un litro ed il costo in lire di un chilometro percorso;

— salvataggio dei dati, da attivare ogni volta che si inseriscono nuovi dati.

Le variabili usate sono le seguenti:

S (13) vettore contenente i soldi spesi in benzina mensilmente;
L (13) vettore contenente i litri di benzina comprati mensilmente;
K (13) vettore contenente i chilometri percorsi ogni mese. In questo vettore saranno contenuti i chilometri letti direttamente dal contachilometri dell'automobile; essi andranno poi confrontati con la variabile F inizializzata precedentemente.

Infine per comodità di stampa le medie sono state raccolte in due vettori: MK (13) e MC (13). Così facendo si sono potute effettuare tutte le stampe necessarie con un solo ciclo di FOR.

Diamo di seguito il diagramma di flusso relativo all'algoritmo suesposto e poi una sua codifica in BASIC.

INIZIO
SALVATAGGIO
VUOI
SALVARE?
W$ = M
SI
RETURN
NO
REGISTRAZIONE
DATI
INIZIO
INSERIMENTO
È VARIATO
IL COSTO
DELLA
BENZINA
W$
SI
AGGIORNAMENTO
VARIABILE
COSTO
NO
QUALE MESE?
BENZINA
MESSA?
IN LIRE
IN LITRI
LIRE
LITRI
INSERIMENTO
COSTO
INSERIMENTO
LITRI
AGGIORNAMENTO
VETTORI
COSTO E
LITRI
AGGIORNAMENTO
VETTORI
COSTO E
LITRI
INPUT
CHILOMETRI
AGGIORNAMENTO
VETTORE
CHILOMETRI
RETURN

```
10   REM CONSUMO AUTOMOBILE
12   REM
14   REM VARIABILI USATE:
16   REM S(13) PER CONTENERE IL COSTO DELLA BENZINA MENSILE
18   REM L(13) PER CONTENERE I LITRI DI BENZINA COMPRATI
20   REM K(13) PER CONTENERE I CHILOMETRI PERCORSI
22   REM
25   DIM S(13): DIM L(13): DIM K(13): DIM MK(13): DIM MC(13)
27   DIM M$(13,12)
30   LET M$(1)=GENNAIO: LET M$(2)=FEBBRAIO: LET M$(3)=MARZO
31   LET M$(4)=APRILE: LET M$(5)=MAGGIO: LET M$(6)=GIUGNO
33   LET M$(7)=LUGLIO: LET M$(8)=AGOSTO: LET M$(9)=SETTEMBRE
35   LET M$(10)=OTTOBRE: LET M$(11)=NOVEMBRE: LET M$(12)=DICEMBRE
37   LET M$(13)=TOTALE
40   LET F=10000
50   LET C=1280
60   CLS
70   PRINT
75   PRINT "========>   M E N U'"
80   PRINT
85   PRINT "1 - INSERIMENTO DATI"
90   PRINT
95   PRINT "2 - CALCOLO MEDIE"
100  PRINT
105  PRINT "3 - SALVATAGGIO DATI"
110  INPUT Z
120  IF Z<0 OR Z>3 THEN GOTO 110
130  GOSUB Z*1000
140  GOTO 60
1000 REM I N S E R I M E N T O  D A T I
1010 CLS
1020 PRINT
1030 PRINT "E' VARIATO IL COSTO DELLA BENZINA RISPETTO A 1280(RISP
ONDI S=SI N=NO)?"
1040 INPUT W$
1050 IF W$<>"N" OR W$<>"S" THEN GOTO 1040
1060 IF W$="N" THEN GOTO 1080
1070 INPUT "NUOVO COSTO DELLA BENZINA";C
1080 INPUT "QUALE MESE";M
1090 CLS
1100 PRINT "MI DICI LA BENZINA IN LIRE O LITRI:"
1110 INPUT W$
1120 IF W$<>"LITRI" OR W$<>"LIRE" THEN GOTO 1110
1130 IF W$="LITRI" THEN GOTO 1200
1140 PRINT
1150 INPUT "QUANTE LIRE DI BENZINA HAI MESSO";A
1160 LET S(M)=S(M)+A
1170 LET B=A/C
1180 LET L(M)=L(M)+B
1190 GOTO 1240
1200 PRINT
1205 INPUT "QUANTI LITRI DI BENZINA HAI MESSO";B
1210 LET L(M)=L(M)+B
1220 LET A=B*C
1230 LET S(M)=S(M)+A
1240 PRINT
```

```
1245 INPUT "A QUANTI CHILOMETRI SEI ARRIVATO";H
1250 LET H=H-F
1260 LET K(M)=H
1270 RETURN
2000 REM S T A M P A  M E D I E
2010 CLS
2020 LET S(13)=0
2030 LET L(13)=0
2040 FOR I=1 TO 12
2050   LET S(13)=S(13)+S(I)
2060   LET L(13)=L(13)+L(I)
2070 NEXT I
2080 LET K(13)=K(M)
2090 REM CALCOLO MEDIE
2100 LET MK(1)=K(1)/L(1)
2110 LET MC(1)=S(1)/K(1)
2120 FOR I=2 TO 12
2130   LET MK(I)=(K(I)-K(I-1))/L(I)
2140   LET MC(I)=S(I)/(K(I)-K(I-1))
2150 NEXT I
2160 LET MK(13)=K(13)/L(13)
2170 LET MC(13)=S(13)/K(13)
2180 REM STAMPE
2190 PRINT
2200 FOR I=1 TO 12
2210   PRINT "MESE DI ";M$(I)
2220   PRINT "CONSUMO IN CHILOMETRI PER LITRO ";MK(I)
2230   PRINT "COSTO DI UN CHILOMETRO LIRE ";MC(I)
2240   PRINT "PREMI <RETURN> PER CONTINUARE"
2250   INPUT W$
2260 NEXT I
2262 PRINT "IN TOTALE HAI PERCORSO ";MK(13);" KM CON UN LITRO"
2266 PRINT
2268 PRINT "OGNI KM TI E' COSTATO ";MC(13); " LIRE"
2270 RETURN
3000 REM R E G I S T R A Z I O N E   D A T I
3010 REM
3020 PRINT "REGISTRAZIONE DATI"
3030 PRINT "PREMI <RETURN> , <M>=MENU'"
3040 INPUT W$
3050 IF W$="M" THEN RETURN
3060 CLS
3070 PRINT "RIAVVOLGI IL NASTRO"
3080 PRINT "QUANDO SEI PRONTO PER REGISTRARE PREMI"
3090 PRINT "<RETURN>"
3100 INPUT W$
3120 SAVE "CONSUMO" LINE 3130
3130 GOTO 60
```

## RICETTARIO DI CUCINA

Quante volte vi sarà capitato di dire, a voi o a vostra moglie: «Avevo trovato quella ricetta; ma ora dove sarà»? E allora comincerete a rovistare tutta la vostra libreria per trovarla.

Questo programma vuole esservi di aiuto per farvi rovistare il meno possibile nella vostra libreria.

Memorizzerà in quattro matrici le ricette e in quattro ulteriori matrici le loro localizzazioni. Le quattro matrici saranno dedicate ai:

— primi piatti,
— secondi piatti,
— contorni,
— dessert.

Il menù conterrà le seguenti opzioni:

— lista, per conoscere tutte le ricette memorizzate. Questa opzione può essere utile per stampare le ricette una volta raggiunta la saturazione della memoria prima di cancellare tutto;

— inserimento, per aggiungere nuove pietanze (*);

— ricerca, per cercare le pietanze di cui desideriamo conoscere la localizzazione;

— azzeramento, per annullare tutte le ricette relative ad un'insieme (primi, secondi, contorni o dessert);

— salvataggio, per memorizzare tutte le nuove pietanze.

È da notare che dopo la scelta fatta con il menù bisogna scegliere su quale insieme operare:

— primi,
— secondi,
— contorni,
— dessert.

Il programma, poi, attiverà le parti relative a ciascun insieme.

Le strutture di dati usate sono:

| | |
|---|---|
| P$: | matrice di primi piatti, |
| L$: | localizzazione dei primi, |
| S$: | matrice dei secondi piatti, |
| M$: | localizzazione dei secondi, |
| C$: | matrice dei contorni, |
| N$: | localizzazione dei contorni, |

(*) I caratteri a disposizione sono:
— 20 caratteri per le pietanze;
— 20 caratteri per la localizzazione.

D$: matrice dei dessert,
O$: localizzazione dei dessert,
U, V, W, Z: contatori per i quattro insiemi.

Le matrici sono dimensionate per 75 pietanze (75 primi, 75 secondi, etc.).

Come al solito si potrà allargare la dimensionalità di esse avendo una memoria RAM maggiore di 16 KByte.

Daremo ora il diagramma di flusso e poi la sua traduzione in BASIC.

INIZIO
LISTA
INPUT
W$
W$ = M
SI
RETURN
NO
CONTATORE
K = 1
K>A
SI
RETURN
NO
LISTA
W$ = M
SI
RETURN
NO
K = K + 1

INIZIO
INSERIMENTO
INCREMENTO
CONTATORE
RELATIVO
CONTATORE
>75
SI
MEMORIA
OCCUPATA
NO
IMMETTI
W$
W$ = M
SI
NO
IMMETTI
I
CAMPI
DEL
RECORD
SI
VUOI
CORREGGERE?
NO
VUOI
ANNULLARE?
SI
DECREMENTO
CONTATORE
A = A—1
RETURN
NO
RETURN

INIZIO RICERCA
IN PIATTI
RELATIVI
VUOI
CERCARE?
IMMETTI W$
W$ = M
SI
RETURN
NO
INPUT
RICETTA
PORTATA
GIUSTA
LUNGHEZZA
RICERCA
RICETTA
NON IN
MEMORIA
SI
CONTAT.
>
N. RICETTE
NO
RETURN
TROVATA
NO
SI
STAMPA
RICETTA
LOCALIZZAZ.
INPUT
W$
RETURN

```
10    REM PROGRAMMA GESTORE DI UN RICETTARIO
20    REM
30    REM DIMENSIONAMENTO DELLE VARIABILI
40    REM PRIMI, SECONDI, CONTORNI, DESSERT
50    REM
60    DIM P$(75,20)
70    DIM S$(75,20)
80    DIM C$(75,20)
90    DIM D$(75,20)
95    DIM L$(75,20)
100   DIM M$(75,20)
105   DIM N$(75,20)
110   DIM O$(75,20)
120   LET U=0
121   LET V=0
122   LET W=0
123   LET Y=0
```

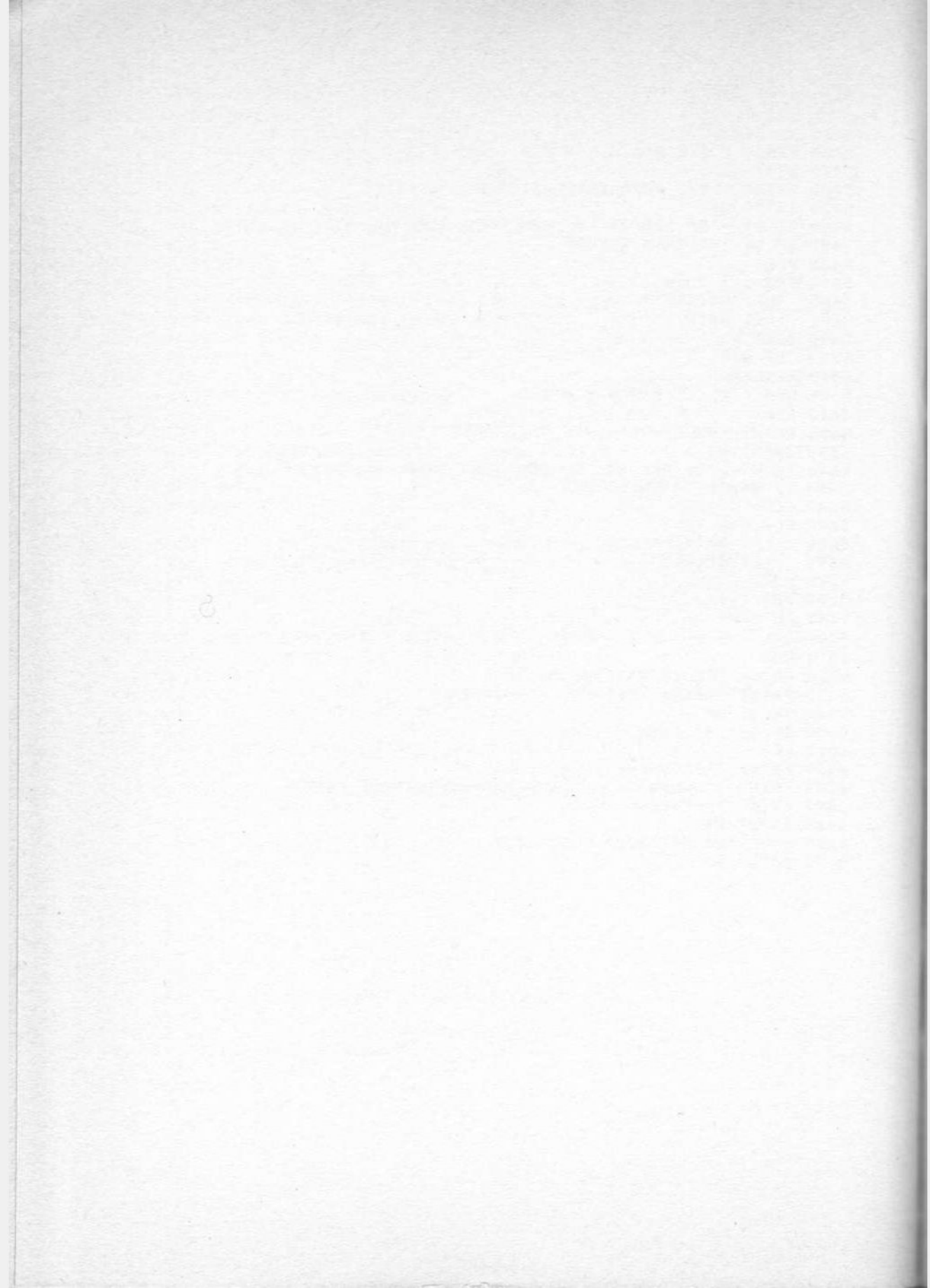
```
130  LET E$='PRIMI'
140  LET F$='SECONDI'
150  LET G$='CONTORNI'
160  LET H$='DESSERT'
180  CLS
190  PRINT ' 1 - LISTA '
200  PRINT ' 2 - INSERIMENTO '
210  PRINT ' 3 - RICERCA '
220  PRINT ' 5 - AZZERAMENTO '
230  PRINT ' 6 - SALVATAGGIO '
240  INPUT 'SCEGLI';Z
250  IF Z<1 OR Z>6 OR Z=4 THEN GOTO 180
252  INPUT 'SCEGLI L'INSIEME DI PIATTI SU CUI VUOI OPERARE';B
254  IF B<1 OR B>4 THEN GOTO 252
260  IF Z=6 THEN GOTO 6000
270  CLS
280  GOSUB Z*1000+(B-1)*200
290  GOTO 180
1000 REM L I S T A
1010 REM
1020 PRINT 'LISTA RICETTARIO'
1030 PRINT 'PREMI <RETURN> , <M> = MENU''
1040 INPUT W$
1050 IF W$='M' THEN RETURN
1060 FOR K=1 TO U
1070   GOSUB 3800+(B-1)*200
1090 NEXT K
1100 RETURN
1200 REM L I S T A
1210 REM
1220 PRINT 'LISTA RICETTARIO'
1230 PRINT 'PREMI <RETURN> , <M> = MENU''
1240 INPUT W$
1250 IF W$='M' THEN RETURN
1260 FOR K=1 TO V
1270   GOSUB 3800+(B-1)*200
1290 NEXT K
1300 RETURN
1400 REM L I S T A
1410 REM
1420 PRINT 'LISTA RICETTARIO'
1430 PRINT 'PREMI <RETURN> , <M> = MENU''
1440 INPUT W$
1450 IF W$='M' THEN RETURN
1460 FOR K=1 TO W
1470   GOSUB 3800+(B-1)*200
1490 NEXT K
1500 RETURN
1600 REM L I S T A
1610 REM
1620 PRINT 'LISTA RICETTARIO'
1630 PRINT 'PREMI <RETURN> , <M> = MENU''
1640 INPUT W$
1650 IF W$='M' THEN RETURN
1660 FOR K=1 TO Y
1670   GOSUB 3800+(B-1)*200
1600 NEXT K
```

```
1700 RETURN
2000 REM I N S E R I M E N T O
2010 REM
2020 LET U=U+1
2030 IF U<75 THEN GOTO 2060
2040 PRINT "NON POSSO INSERIRE: MEMORIA PIENA!"
2050 GOTO 2194
2060 PRINT "INSERIMENTO DATI"
2070 PRINT "PREMI <RETURN> , <M> = MENU''"
2080 INPUT W$
2090 IF W$="M" THEN GOTO 2194
2100 CLS
2110 PRINT "INSERISCI I DATI COME SEGUE"
2120 PRINT E$
2130 PRINT "LOCALIZZAZIONE"
2140 INPUT P$(U)
2150 PRINT P$(U)
2160 INPUT L$(U)
2170 PRINT L$(U)
2175 PRINT "VUOI CORREGGERE? (S/N)"
2177 INPUT W$
2180 IF W$<>"S" THEN GOTO 2188
2184 CLS
2186 GOTO 2110
2188 PRINT "VUOI ANNULLARE? <S/N>"
2190 INPUT W$
2192 IF W$="N" OR W$="" THEN RETURN
2194 LET U=U-1
2196 RETURN
2200 REM I N S E R I M E N T O
2210 REM
2220 LET V=V+1
2230 IF V<75 THEN GOTO 2260
2240 PRINT "NON POSSO INSERIRE: MEMORIA PIENA!"
2250 GOTO 2394
2260 PRINT "INSERIMENTO DATI"
2270 PRINT "PREMI <RETURN> , <M> = MENU''"
2280 INPUT W$
2290 IF W$="M" THEN GOTO 2394
2300 CLS
2310 PRINT "INSERISCI I DATI COME SEGUE"
2320 PRINT F$
2330 PRINT "LOCALIZZAZIONE"
2340 INPUT S$(V)
2350 PRINT S$(V)
2360 INPUT M$(V)
2370 PRINT M$(V)
2375 PRINT "VUOI CORREGGERE? (S/N)"
2377 INPUT W$
2380 IF W$<>"S" THEN GOTO 2388
2384 CLS
2386 GOTO 2310
2388 PRINT "VUOI ANNULLARE? <S/N>"
2390 INPUT W$
2392 IF W$="N" OR W$="" THEN RETURN
2394 LET V=V-1
2396 RETURN
```

```
2400 REM I N S E R I M E N T O
2410 REM
2420 LET W=W+1
2430 IF W<75 THEN GOTO 2460
2440 PRINT "NON POSSO INSERIRE: MEMORIA PIENA!"
2450 GOTO 2594
2460 PRINT "INSERIMENTO DATI"
2470 PRINT "PREMI <RETURN> , <M> = MENU'"
2480 INPUT W$
2490 IF W$="M" THEN GOTO 2594
2500 CLS
2510 PRINT "INSERISCI I DATI COME SEGUE"
2520 PRINT G$
2530 PRINT "LOCALIZZAZIONE"
2540 INPUT C$(W)
2550 PRINT C$(W)
2560 INPUT N$(W)
2570 PRINT N$(W)
2575 PRINT "VUOI CORREGGERE? (S/N)"
2577 INPUT W$
2580 IF W$<>"S" THEN GOTO 2588
2584 CLS
2586 GOTO 2510
2588 PRINT "VUOI ANNULLARE? <S/N>"
2590 INPUT W$
2592 IF W$="N" OR W$="" THEN RETURN
2594 LET W=W-1
2596 RETURN
2600 REM I N S E R I M E N T O
2610 REM
2620 LET Y=Y+1
2630 IF Y<75 THEN GOTO 2660
2640 PRINT "NON POSSO INSERIRE: MEMORIA PIENA!"
2650 GOTO 2794
2660 PRINT "INSERIMENTO DATI"
2670 PRINT "PREMI <RETURN> , <M> = MENU'"
2680 INPUT W$
2690 IF W$="M" THEN GOTO 2794
2700 CLS
2710 PRINT "INSERISCI I DATI COME SEGUE"
2720 PRINT H$
2730 PRINT "LOCALIZZAZIONE"
2740 INPUT D$(Y)
2750 PRINT D$(Y)
2760 INPUT O$(Y)
2770 PRINT O$(Y)
2775 PRINT "VUOI CORREGGERE? (S/N)"
2777 INPUT W$
2780 IF W$<>"S" THEN GOTO 2788
2784 CLS
2786 GOTO 2710
2788 PRINT "VUOI ANNULLARE? <S/N>"
2790 INPUT W$
2792 IF W$="N" OR W$="" THEN RETURN
2794 LET Y=Y-1
2796 RETURN
3000 REM R I C E R C A
```

```
3010 REM
3020 CLS
3030 PRINT 'RICERCA'
3040 PRINT 'PREMI <RETURN> , <M>=MENU''
3045 INPUT W$
3050 IF W$='M' THEN RETURN
3100 PRINT 'INSERISCI IL PRIMO DA RICERCARE'
3105 INPUT B$
3110 IF LEN (B$)>20 THEN GOTO 3105
3115 IF LEN (B$)=20 THEN GOTO 3160
3120 LET B$=B$+' '
3125 GOTO 3115
3160 FOR K=1 TO U
3165   IF P$(K)=B$ THEN GOTO 3800+(B-1)*200
3170 NEXT K
3175 GOTO 3860
3200 REM R I C E R C A
3210 REM
3220 CLS
3230 PRINT 'RICERCA'
3240 PRINT 'PREMI <RETURN> , <M>=MENU''
3245 INPUT W$
3250 IF W$='M' THEN RETURN
3300 PRINT 'INSERISCI IL SECONDO DA RICERCARE'
3305 INPUT B$
3310 IF LEN (B$)>20 THEN GOTO 3305
3315 IF LEN (B$)=20 THEN GOTO 3360
3320 LET B$=B$+' '
3325 GOTO 3315
3360 FOR K=1 TO V
3365   IF S$(K)=B$ THEN GOTO 3800+(B-1)*200
3370 NEXT K
3375 GOTO 3860
3400 REM R I C E R C A
3410 REM
3420 CLS
3430 PRINT 'RICERCA'
3440 PRINT 'PREMI <RETURN> , <M>=MENU''
3445 INPUT W$
3450 IF W$='M' THEN RETURN
3500 PRINT 'INSERISCI IL CONTORNO DA RICERCARE'
3505 INPUT B$
3510 IF LEN (B$)>20 THEN GOTO 3505
3515 IF LEN (B$)=20 THEN GOTO 3560
3520 LET B$=B$+' '
3525 GOTO 3515
3560 FOR K=1 TO W
3565   IF C$(K)=B$ THEN GOTO 3800+(B-1)*200
3570 NEXT K
3575 GOTO 3860
3600 REM R I C E R C A
3610 REM
3620 CLS
3630 PRINT 'RICERCA'
3640 PRINT 'PREMI <RETURN> , <M>=MENU''
3645 INPUT W$
3650 IF W$='M' THEN RETURN
```

```
3700 PRINT "INSERISCI IL DESSERT DA RICERCARE"
3705 INPUT B$
3710 IF LEN (B$)>20 THEN GOTO 3705
3715 IF LEN (B$)=20 THEN GOTO 3760
3720 LET B$=B$+" "
3725 GOTO 3715
3760 FOR K=1 TO Y
3765   IF D$(K)=B$ THEN GOTO 3800+(B-1)*200
3770 NEXT K
3775 GOTO 3860
3800 CLS
3802 PRINT "IL PIATTO ";P$(K);"SI TROVA ";L$(K)
3815 PRINT "PER CONTINUARE PREMI <RETURN>"
3820 INPUT W$
3840 RETURN
3860 CLS
3870 PRINT "PIATTO NON PRESENTE"
3880 RETURN
4000 CLS
4002 PRINT "IL PIATTO ";S$(K);"SI TROVA ";M$(K)
4015 PRINT "PER CONTINUARE PREMI <RETURN>"
4020 INPUT W$
4040 RETURN
4200 CLS
4202 PRINT "IL PIATTO ";C$(K);"SI TROVA ";N$(K)
4215 PRINT "PER CONTINUARE PREMI <RETURN>"
4220 INPUT W$
4240 RETURN
4400 CLS
4402 PRINT "IL PIATTO ";D$(K);"SI TROVA ";O$(K)
4415 PRINT "PER CONTINUARE PREMI <RETURN>"
4420 INPUT W$
4440 RETURN
5000 REM A Z Z E R A M E N T O
5010 REM
5020 PRINT "VUOI VERAMENTE AZZERARE? (S/N)"
5030 INPUT W$
5040 IF W$<>"S" AND W$<>N THEN GOTO 5030
5050 IF W$="N" THEN RETURN
5060 CLS
5070 FOR K=1 TO U
5080   LET P$(K)=""
5090   LET L$(K)=""
5140 NEXT K
5150 LET U=0
5160 RETURN
5200 REM A Z Z E R A M E N T O
5210 REM
5220 PRINT "VUOI VERAMENTE AZZERARE? (S/N)"
5230 INPUT W$
5240 IF W$<>"S" AND W$<>N THEN GOTO 5230
5250 IF W$="N" THEN RETURN
5260 CLS
5270 FOR K=1 TO V
5280   LET S$(K)=""
5290   LET M$(K)=""
```

```
5340 NEXT K
5350 LET V=0
5360 RETURN
5400 REM A Z Z E R A M E N T O
5410 REM
5420 PRINT "VUOI VERAMENTE AZZERARE? (S/N)"
5430 INPUT W$
5440 IF W$<>"S" AND W$<>N THEN GOTO 5430
5450 IF W$="N" THEN RETURN
5460 CLS
5470 FOR K=1 TO W
5480   LET C$(K)=""
5490   LET N$(K)=""
5540 NEXT K
5550 LET W=0
5560 RETURN
5600 REM A Z Z E R A M E N T O
5610 REM
5620 PRINT "VUOI VERAMENTE AZZERARE? (S/N)"
5630 INPUT W$
5640 IF W$<>"S" AND W$<>N THEN GOTO 5630
5650 IF W$="N" THEN RETURN
5660 CLS
5670 FOR K=1 TO Y
5680   LET D$(K)=""
5690   LET O$(K)=""
5740 NEXT K
5750 LET Y=0
5760 RETURN
6000 REM R E G I S T R A Z I O N E   D A T I
6010 REM
6020 PRINT "REGISTRAZIONE DATI"
6030 PRINT "PREMI <RETURN> , <M>=MENU'"
6040 INPUT W$
6050 IF W$="M" THEN RETURN
6060 CLS
6070 PRINT "RIAVVOLGI IL NASTRO"
6080 PRINT "QUANDO SEI PRONTO PER REGISTRARE PREMI"
6090 PRINT "<RETURN>"
6100 INPUT W$
6120 SAVE "RICETTARIO" LINE 6130
6130 GOTO 180
```

# 4. PROGRAMMI DI UTILITÀ PER L'UOMO D'AFFARI

## GESTIONE CLIENTI

Il seguente programma mette a disposizione dell'utilizzatore un metodo per gestire i pagamenti e le loro scadenze, per un certo numero di clienti.

Una volta che si sono caricate le rate che dovranno essere pagate dai clienti, mese per mese, si potrà lavorare sui dati contenuti sia controllando le scadenze, che aggiornando i pagamenti.

Il programma funziona a lista, le cui opzioni sono:

— lista di dati memorizzati, che emetterà tutti i clienti con la relativa situazione;

— inserimento dati, per inizializzare tutte le operazioni inserendo clienti e relative scadenze;

— ricerca di record, che può essere effettuata in base al nome e cognome del cliente, o al mese che si vuole analizzare;

— pagamenti, con il quale si aggiorneranno i vari pagamenti dei clienti (aggiornerà sia i pagamenti mensili che il totale pagato);

— azzeramento, per azzerare tutte le variabili alla fine di ogni anno;

— salvataggio dati, da attivare ogni volta che si sono manipolati i dati memorizzati.

Le strutture di dati si basano su tre matrici:

C$: contenente i cognomi dei clienti;
N$: contenente i nomi di clienti;
MESI: contenente le rate da versare, da parte del cliente, ogni mese; il totale che dovrà essere pagato durante l'anno e il totale già pagato dal cliente stesso.

Diamo ora il solito diagramma di flusso dell'algoritmo e poi la sua traduzione nel linguaggio BASIC.

INIZIO
SCELTA OPZIONE
LISTA
INSERIMENTO DATI
RICERCA RECORD
AZZERAMENTO
AGGIORN. PAGAMENTI
SALVATAGGIO DATI
INIZIO LISTA
INPUT W$
W$ = M
SI
RETURN
NO
CONTATORE K = 1
K>A
SI
RETURN
NO
LISTA
W$ = M
SI
RETURN
NO
K = K + 1

INIZIO
INSERIMENTO
INCREMENTO
CONTATORE
A = A + 1
A > 100
SI
MEMORIA
OCCUPATA
NO
IMMETTI
W$
W$ = M
SI
NO
IMMETTI
I CAMPI
DEL
RECORD
SOMMATORIA
MESI
INSERITA A
MESI (A,13)
SI
VUOI
CORREGGERE?
NO
VUOI
ANNULLARE?
SI
DECREMENTO
CONTATORE
A = A—1
RETURN
NO
RETURN

INIZIO
RICERCA
CONTATORE
C = 0
VUOI
CERCARE?
IMMETTI
W$
W$ = M
SI
RETURN
INPUT
OPZIONE
RICERCA
OPZIONE
RICERCA
INPUT
COGNOME
NOME
CLIENTE
INPUT
COGNOME
CLIENTE
INPUT
MESE
PORTATI LUNG.
GIUSTA
RICERCA
NOME
PORTATO
LUNGHEZZA
GIUSTA
RICERCA
NOME
RICERCA
CLIENTI
CHE DEVONO
PAGARE
SI
3
CONTATORE
K > A
NO
TROVATO
NO
SI
1
SI
2
CONTATORE
K>A
NO
TROVATO
NO
SI
GOSUB
1

①
VISUALIZZAZIONE
STAMPA RECORD
INPUT W$
CONTATORE C INCREMENTATO RETURN
②
CONTATORE>0
SI
RETURN
NO
NOMINATIVO NON IN MEMORIA
③
RETURN
INIZIO AGGIORNAMENTO PAGAMENTI
IMMETTI COGNOME E NOME
RICERCA SE CLIENTE ESISTE
ESISTE
NO
STAMPA CLIENTE NON ESISTE
RETURN
SI
MESE DI PAGAMENTO
HA GIÀ PAGATO
SI
NON HA NULLA DA PAGARE
RETURN
NO
SOMMA PAGATA
AGGIORNAMENTO MESE E TOTALE PAGATO
RETURN

```
10   REM PROGRAMMA GESTORE DI UN ELENCO DI CLIENTI
20   REM
30   REM DIMENSIONAMENTO DELLE VARIABILI
40   REM COGNOME, NOME, PAGAMENTI MESI
50   REM
60   DIM C$(100,15)
70   DIM N$(100,10)
90   DIM MESI(100,14)
120  LET A=0
130  LET L$='COGNOME'
140  LET D$='NOME'
180  CLS
190  PRINT ' 1 - LISTA '
200  PRINT ' 2 - INSERIMENTO '
210  PRINT ' 3 - RICERCA '
220  PRINT ' 4 - PAGAMENTI '
230  PRINT ' 5 - AZZERAMENTO '
240  PRINT ' 6 - SALVATAGGIO '
250  INPUT 'SCEGLI';Z
260  IF Z<1 OR Z>6 THEN GOTO 180
270  CLS
280  GOSUB Z*1000
290  GOTO 180
1000 REM L I S T A
1010 REM
1020 PRINT 'LISTA CLIENTI'
1030 PRINT 'PREMI <RETURN> , <M> = MENU''
1040 INPUT W$
```

```
1050 IF W$='M' THEN RETURN
1060 FOR K=1 TO A
1070   GOSUB 3600
1090 NEXT K
1100 RETURN
2000 REM I N S E R I M E N T O
2010 REM
2020 LET A=A+1
2030 IF A<100 THEN GOTO 2060
2040 PRINT 'NON POSSO INSERIRE: MEMORIA PIENA!'
2050 GOTO 2370
2060 PRINT 'INSERIMENTO DATI'
2070 PRINT 'PREMI <RETURN> , <M> = MENU''
2080 INPUT W$
2090 IF W$='M' THEN GOTO 2370
2100 CLS
2110 PRINT 'INSERISCI I DATI COME SEGUE'
2120 PRINT L$
2130 PRINT D$
2140 PRINT 'I PAGAMENTI DA EFFETTUARE NEI DODICI MESI'
2170 INPUT C$(A)
2180 PRINT C$(A)
2190 INPUT N$(A)
2200 PRINT N$(A)
2210 FOR I=1 TO 12
2220   INPUT MESI(A,I)
2230   PRINT MESI(A,I)
2240 NEXT I
2245 LET MESI(A,14)=0
2250 LET SOM=0
2255 FOR I=1 TO 12
2260   LET SOM=SOM+MESI(A,I)
2265 NEXT I
2280 LET MESI(A,13)=SOM
2290 PRINT 'VUOI CORREGGERE? (S/N)'
2300 INPUT W$
2310 IF W$<>'S' THEN GOTO 2340
2320 CLS
2330 GOTO 2110
2340 PRINT 'VUOI ANNULLARE? <S/N>'
2350 INPUT W$
2360 IF W$='N' OR W$='' THEN RETURN
2370 LET A=A-1
2380 RETURN
3000 REM R I C E R C A
3010 REM
3020 LET C=0
3030 PRINT 'RICERCA'
3040 PRINT 'PREMI <RETURN> , <M>=MENU''
3045 INPUT W$
3050 IF W$='M' THEN RETURN
3055 CLS
3060 PRINT 'RICERCA CON:'
3062 PRINT '1 - COGNOME E NOME'
3064 PRINT '2 - COGNOME'
3066 PRINT '3 - MESE'
3072 PRINT 'SCEGLI?'
```

```
3075 INPUT Z
3080 IF Z<1 OR Z>3 THEN GOTO 3075
3085 CLS
3090 GOSUB Z*100+3000
3095 RETURN
3100 PRINT 'INSERISCI IL COGNOME'
3105 INPUT B$
3110 IF LEN (B$)>15 THEN GOTO 3105
3115 IF LEN (B$)=15 THEN GOTO 3130
3120 LET B$=B$+' '
3125 GOTO 3115
3130 PRINT 'INSERISCI IL NOME'
3135 INPUT E$
3140 IF LEN (E$)>10 THEN GOTO 3135
3145 IF LEN (E$)=10 THEN GOTO 3160
3150 LET E$=E$+' '
3155 GOTO 3145
3160 FOR K=1 TO A
3165   IF C$(K)=B$ AND N$(K)=E$ THEN GOTO 3600
3170 NEXT K
3175 GOTO 3660
3200 PRINT 'INSERISCI IL COGNOME'
3205 INPUT B$
3210 IF LEN (B$)>15 THEN GOTO 3205
3215 IF LEN (B$)=15 THEN GOTO 3230
3220 LET B$=B$+' '
3225 GOTO 3215
3230 FOR K=1 TO A
3240   IF C$(K)=B$ THEN GOSUB 3600
3250 NEXT K
3260 GOTO 3650
3300 PRINT 'INSERISCI IL MESE CON UN NUMERO DA 1 A 12'
3305 INPUT B
3330 FOR K=1 TO A
3340   IF MESI(K,B)<>0 THEN GOSUB 3700
3350 NEXT K
3360 GOTO 3750
3600 CLS
3602 PRINT L$;' ';C$(K)
3604 PRINT D$;' ';N$(K)
3606 PRINT 'IMPORTI DA PAGARE NEI DODICI MESI SCRITTI CONSECUTIVAM
ENTE'
3608 PRINT MESI(K,1),MESI(K,2),MESI(K,3),MESI(K,4),MESI(K,5),MESI(
K,6),
3610 PRINT MESI(K,7),MESI(K,8),MESI(K,9),MESI(K,10),MESI(K,11),MES
I(K,12)
3612 PRINT 'TOTALE DA PAGARE', MESI(K,13)
3614 PRINT 'TOTALE GIA' PAGATO', MESI(K,14)
3615 PRINT 'PER CONTINUARE PREMI <RETURN>'
3620 INPUT W$
3630 LET C=C+1
3640 RETURN
3650 IF C>0 THEN RETURN
3660 CLS
3670 PRINT 'RECORD NON PRESENTE'
3680 RETURN
3700 CLS
3702 PRINT 'IL CLIENTE ';C$(K),N$(K)
```

```
3705 PRINT
3710 PRINT "DEVE PAGARE NEL MESE RICHIESTO",MESI(K,B)
3715 PRINT "PER CONTINUARE PREMI <RETURN>"
3720 INPUT W$
3730 LET C=C+1
3740 RETURN
3750 IF C>0 THEN RETURN
3760 CLS
3770 PRINT "NON ESISTE ALCUN CLIENTE CHE DEVE PAGARE NEL MESE RICH
IESTO"
3780 RETURN
4000 REM A G G I O R N A M E N T O   P A G A M E N T I
4010 CLS
4020 PRINT "INSERISCI IL COGNOME"
4030 INPUT B$
4040 IF LEN (B$)>15 THEN GOTO 4030
4045 IF LEN (B$)=15 THEN GOTO 4080
4050 LET B$=B$+" "
4060 GOTO 4045
4080 PRINT "INSERISCI IL NOME"
4090 INPUT E$
4100 IF LEN (E$)>10 THEN GOTO 4090
4115 IF LEN (E$)=10 THEN GOTO 4140
4120 LET E$=E$+" "
4125 GOTO 4115
4140 FOR K=1 TO A
4150   IF C$(K)=B$ AND N$(K)=E$ THEN GOTO 4190
4160 NEXT K
4170 PRINT "NON ESISTE NESSUN CLIENTE CON QUESTO NOME"
4180 RETURN
4190 INPUT "QUALE MESE HA PAGATO (NUMERO DA 1 A 12)?";W
4200 IF W<1 OR W>12 THEN GOTO 4190
4210 IF MESI(K,W)<>0 THEN GOTO 4240
4220 PRINT "IL CLIENTE";C$(K),N$(K);"NON AVEVA NULLA DA PAGARE"
4230 RETURN
4240 INPUT "QUALE CIFRA HA PAGATO?"T
4250 IF T>MESI(K,W) THEN GOTO 4240
4260 LET MESI(K,W)=MESI(K,W)-T
4270 LET MESI(K,14)=MESI(K,14)+T
4280 RETURN
5000 REM A Z Z E R A M E N T O
5010 REM
5020 PRINT "VUOI VERAMENTE AZZERARE? (S/N)"
5030 INPUT W$
5040 IF W$<>"S" AND W$<>N THEN GOTO 5030
5050 IF W$="N" THEN RETURN
5060 CLS
5070 FOR K=1 TO A
5080   LET C$(K)=""
5090   LET N$(K)=""
5100   FOR I=1 TO 14
5110     LET MESI(K,I)=0
5120   NEXT I
5140 NEXT K
5150 LET A=0
5160 RETURN
6000 REM R E G I S T R A Z I O N E   D A T I
```

```
6010 REM
6020 PRINT "REGISTRAZIONE DATI"
6030 PRINT "PREMI <RETURN> , <M>=MENU'"
6040 INPUT W$
6050 IF W$="M" THEN RETURN
6060 CLS
6070 PRINT "RIAVVOLGI IL NASTRO"
6080 PRINT "QUANDO SEI PRONTO PER REGISTRARE PREMI"
6090 PRINT "<RETURN>"
6100 INPUT W$
6120 SAVE "CLIENTI" LINE 6130
6130 GOTO 180
```

## CALCOLO DEGLI INTERESSI BANCARI

Vi può essere capitato, talvolta, di desiderar di conoscere il valore che un capitale che possedete assumerà tra un numero definito di anni, ad un tasso d'interesse costante.

L'algoritmo per effettuare questo calcolo è molto semplice poiché si basa sulla formula seguente:

$$A = C\,(1 + i)^{Y}$$

dove **C** rappresenta il valore iniziale del capitale, **i** il tasso di interesse relativo ed **Y** il numero di anni di deposito.

Nel programma seguente vengono date diverse tabelle (che volendo potreste anche stamparvi da soli, sostituendo ai PRINT di riga 120, 1100, 1110 le istruzioni relative), che contengono i valori assunti dal vostro capitale, in anni consecutivi e a un determinato tasso d'interesse.

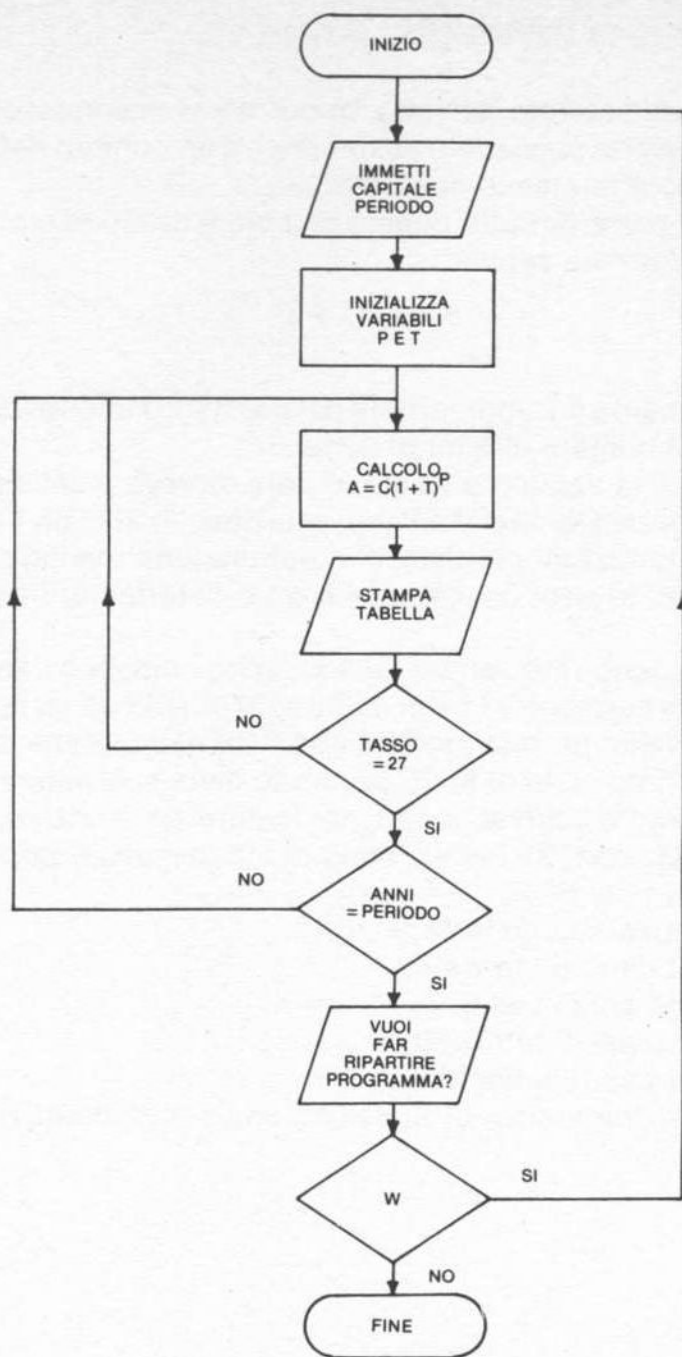
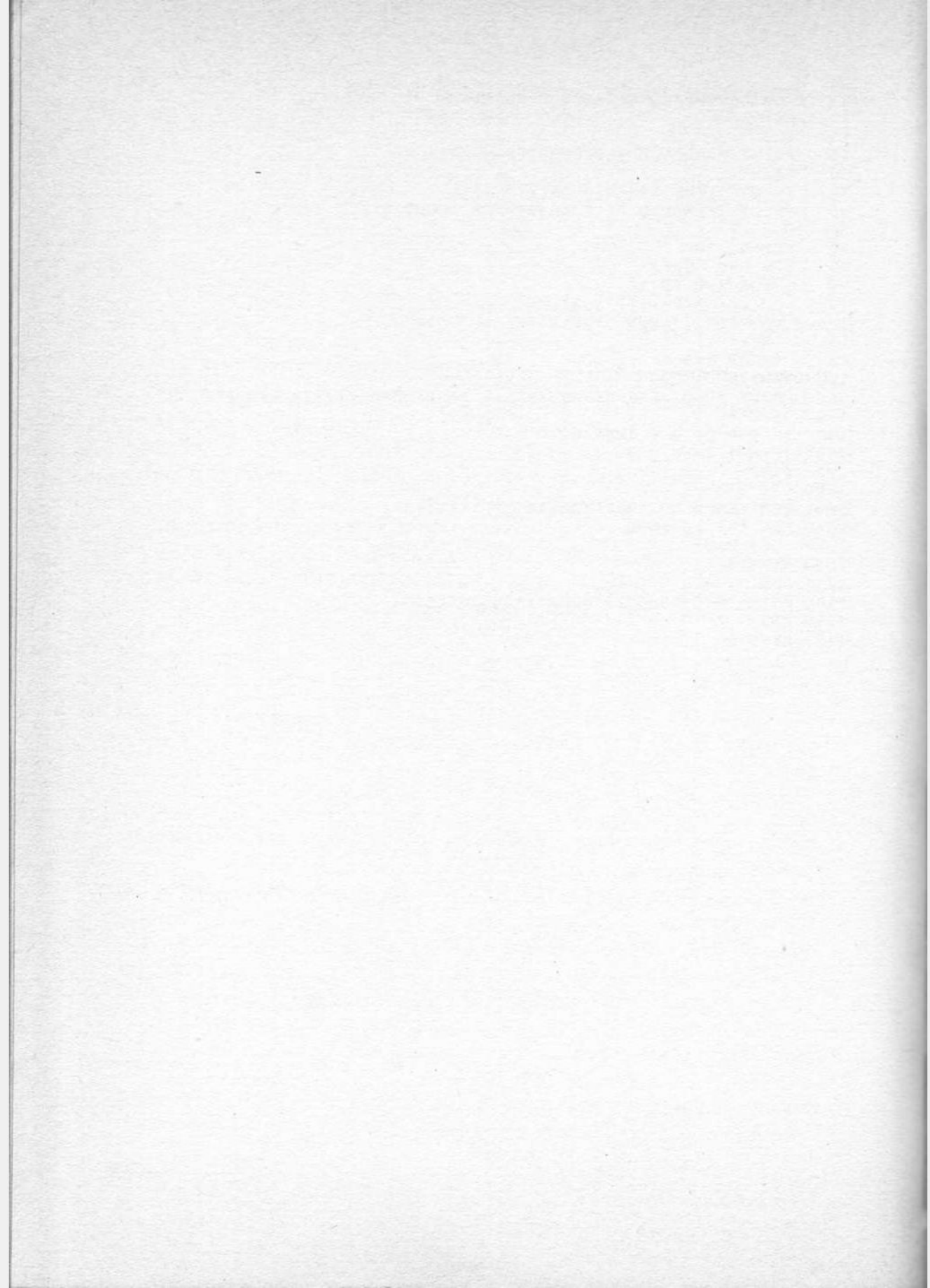
Gli anni vengono fatti variare da 1 all'anno immesso dall'utilizzatore del programma, mentre i tassi dall'8 al 27% (se non siete soddisfatti di questi insiemi potrete modificare i FOR esistenti nella codifica).

Tramite un finto ciclo di FOR, esistente nella subroutine 1000, si fa attendere il quadro sullo schermo, per facilitarne la lettura, prima della sua cancellazione. Se i tempi sono lunghi o corti, si potrà agire sul limite maggiore del FOR.

Le variabili usate sono le seguenti:

- C: per il capitale iniziale,
- P1: per gli anni in esame,
- T: per i tassi di interesse,
- A: per il capitale aggiornato.

Diamo ora il diagramma di flusso ed una sua codifica nel linguaggio BASIC.

INIZIO
IMMETTI
CAPITALE
PERIODO
INIZIALIZZA
VARIABILI
P E T
CALCOLO
A = C(1 + T)P
STAMPA
TABELLA
NO
TASSO
= 27
SI
NO
ANNI
= PERIODO
SI
VUOI
FAR
RIPARTIRE
PROGRAMMA?
SI
W
NO
FINE

```
5     REM CALCOLO INTERESSI BANCARI
7     REM
10    CLS
15    PRINT "CALCOLO INTERESSI BANCARI"
20    PRINT
30    INPUT "INVESTIMENTO INIZIALE";C
40    INPUT "PERIODO DA CONSIDERARE (ANNI)";P1
50    CLS
60    GOSUB 1100
80    FOR P=1 TO P1
90      FOR T=8 TO 27
100       LET A=C*(1+T/100) P
120       PRINT "ANNO  ";P,T;"%","    ";A
130     NEXT T
140    GOSUB 1000
150   NEXT P
160   PRINT "VUOI FAR RIPARTIRE IL PROGRAMMA (1=SI, 0=NO)"
170   INPUT W
180   IF W<0 OR W>1 THEN GOTO 170
190   IF W=1 THEN GOTO 10
200   STOP
1000 LET U=0
1001 REM TEMPO DA CALIBRARE!!!!!!!!!!!!!!
1010 FOR I=1 TO 1000
1020  LET U=U+1
1030 NEXT I
1040 CLS
1100 PRINT "PERIODO","TASSO","AMMONTARE"
1110 PRINT "-------","-----","---------"
1120 RETURN
```

# 5. PROGRAMMI DI GIOCHI VARI

## DISCESA SULLA LUNA

Questo è il primo programma di giochi che viene presentato in questo libro. Tale gioco è «dinamico»: vale a dire che l'elaboratore risponde alla persona che lo sta usando dando delle informazioni sull'andamento del gioco in quel preciso istante. A seconda delle decisioni dell'uomo il gioco prenderà una direzione diversa da quella che avrebbe intrapreso con qualsiasi altro comando umano.

Si tenga comunque presente che i giochi qui proposti come dinamici non manifesteranno tale dinamicità sullo schermo video: non comprendono cioè quella parte del programma che sfrutta le informazioni per far muovere delle figure sullo schermo.

Vi saranno poche difficoltà a sfruttare le informazioni date di volta in volta dall'elaboratore per costruire delle figure che renderanno il gioco più piacevole.

Nel caso nostro, il rapporto elaboratore utente si basa soltanto su informazioni scritte che l'elaboratore invia per informare dell'andamento del gioco.

Vediamo comunque il programma in oggetto. Lo scopo sarà di effettuare un atterraggio morbido sulla luna, avendo a disposizione 15.000 litri di carburante e partendo da un'altezza di 100 Km con una velocità di 100 m/sec, verso la luna.

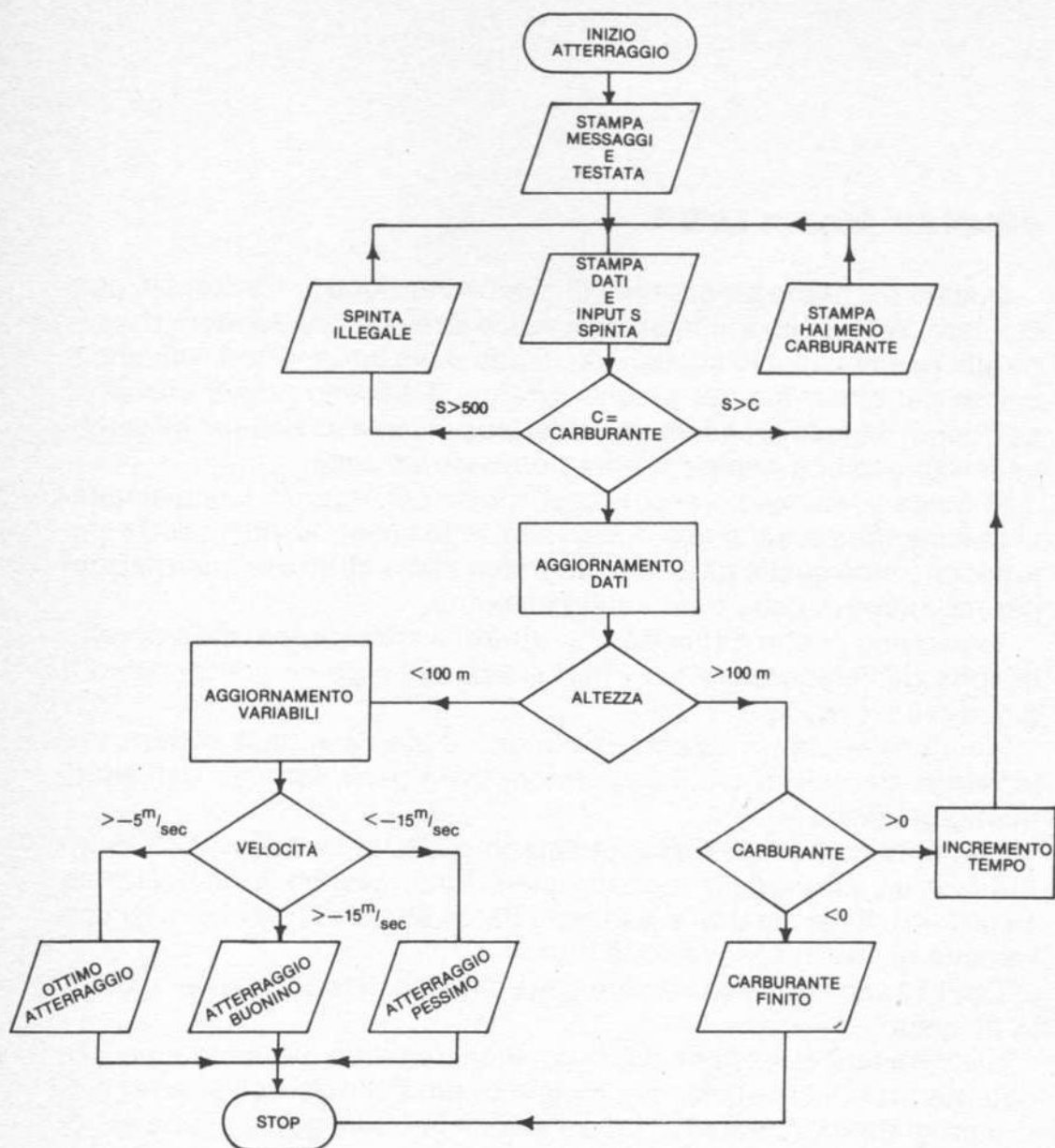
Ogni 10 secondi si dovrà dare una spinta con i retrorazzi per frenare la discesa.

Si dovrà fare attenzione a non consumare molto carburante per poi poterne avere abbastanza nel momento più delicato dell'atterraggio.

Il programma vi dirà alla fine se siete adatti alla guida di una astronave o se proprio vi dovrete rinunciare.

Non vi sono strutture di dati complesse, ma si agisce solo su singole variabili.

Daremo al solito il diagramma di flusso con la sua traduzione in linguaggio BASIC.

INIZIO
ATTERRAGGIO
STAMPA
MESSAGGI
E
TESTATA
STAMPA
DATI
E
INPUT S
SPINTA
SPINTA
ILLEGALE
STAMPA
HAI MENO
CARBURANTE
S>500
C =
CARBURANTE
S>C
AGGIORNAMENTO
DATI
<100 m
ALTEZZA
>100 m
AGGIORNAMENTO
VARIABILI
>−5 m/sec
VELOCITÀ
<−15 m/sec
>−15 m/sec
CARBURANTE
>0
INCREMENTO
TEMPO
<0
OTTIMO
ATTERRAGGIO
ATTERRAGGIO
BUONINO
ATTERRAGGIO
PESSIMO
CARBURANTE
FINITO
STOP

```
10  REM
20  REM DISCESA LUNARE
30  REM
40  REM VARIABILI USATE
50  REM   H   ALTEZZA
60  REM   V   VELOCITA'
70  REM   E   CARBURANTE
80  REM   T   TEMPO
90  REM
100 PRINT"VI SIETE DISTACCATI DALL'ASTRONAVE MADRE"
110 PRINT"E STATE DISCENDENDO SULLA LUNA"
120 PRINT"DOPO IL PROMPT -?- SCRIVETE IL VALORE DELLA"
130 PRINT"SPINTA DEI RETRORAZZI CHE DEVE VARIARE DA  "
140 PRINT"0 A 500"
150 PRINT"----------- GOOD LUCK -----------"
160 PRINT
170 PRINT"TEMPO  VELOC. ALT.  CARB.  SPINTA"
180 PRINT" SEC    MT/S   KM    LB    NWT/100"
190 PRINT
200 LET H=100
210 LET S=5000
220 LET E=15000
230 LET T=0
240 LET G=2.4
250 LET V=-100
260 PRINT T;TAB(7);V;TAB(13);(INT(H*10))/10;TAB(18);E;TAB(20);"F=";
270 INPUT F
275 REM
277 REM CONTROLLO SPINTA
279 REM
280 IF ABS(F-250)>251 THEN 320
290 IF F>E THEN 340
300 GOTO 360
305 REM
310 REM SPINTA ILLEGALE
315 REM
320 PRINT "SPINTA ILLEGALE PREGO RIPETERE"
330 GOTO 260
332 REM
334 REM SPINTA TROPPO GROSSA PER IL CARBURANTE CHE HAI
336 REM
340 PRINT "SONO RIMASTE";E;"UNITA' DI CARBURANTE"
350 GOTO 260
360 LET E=E-F
370 LET V1=V
380 LET V2=2000*F/(S+E)-10*G
390 LET V=V+V2
400 LET D=(V1+V2/2)*10
410 LET H=H+D/1000
420 IF H=<0.01 THEN 470
430 IF E=< 0 THEN 460
440 LET T=T+10
450 GOTO 260
452 REM
454 REM CARBURANTE FINITO
456 REM
460 PRINT "FINITO IL CARBURANTE";H;"KM PIU' IN SU"
```

```
465 GOTO 700
470 LET F=0
480 LET K=0
490 LET K=k +0.1
500 LET V1=V
510 LET V2=20*F/(S+E)-0.1*G
520 LET V=V+V2
530 LET H=H+(V1+V2/2)/1000
540 IF H>0 THEN 490
542 REM
544 REM STIAMO ATTERRANDO
546 REM
550 PRINT 'IMPATTO FRA CIRCA';K;'SECONDI'
560 PRINT 'ATTENDI'
561 PRINT
562 PRINT
570 IF V=>-5 THEN 630
580 IF V=>-15 THEN 660
582 REM
584 REM TROPPA VELOCITA'
586 REM
590 PRINT 'SBOOM!!! VELOCITA' = ';V;'M/SEC'
600 PRINT 'CIOE' ';V*3.2*3600/5280;'MPH-- ACCIPICCHIA'
610 PRINT 'TI SEI SFRACELLATO !!!'
620 GOTO 700
622 REM
624 REM ATTERRAGGIO OTTIMO
626 REM
630 PRINT 'OTTIMO ATTERRAGGIO V=';V*3.2*3600/5280;'MPH'
640 PRINT 'AVEVI';E;'KG DI CARBURANTE'
650 GOTO 700
652 REM
654 REM ATTERRAGGIO BUONINO
656 REM
660 PRINT
670 PRINT 'V =';V*3.2*3600/5280;'MPH'
680 PRINT 'ATTERRAGGIO BUONINO,  SEI ANCORA VIVO"
700 END
```

## MASTER MIND

Questo gioco è diventato ormai famosissimo. Lo scopo è quello di indovinare con un numero finito di mosse, nel nostro caso 10 (vedi il controllo della variabile T nell'istruzione 500), un numero pensato dal vostro elaboratore.

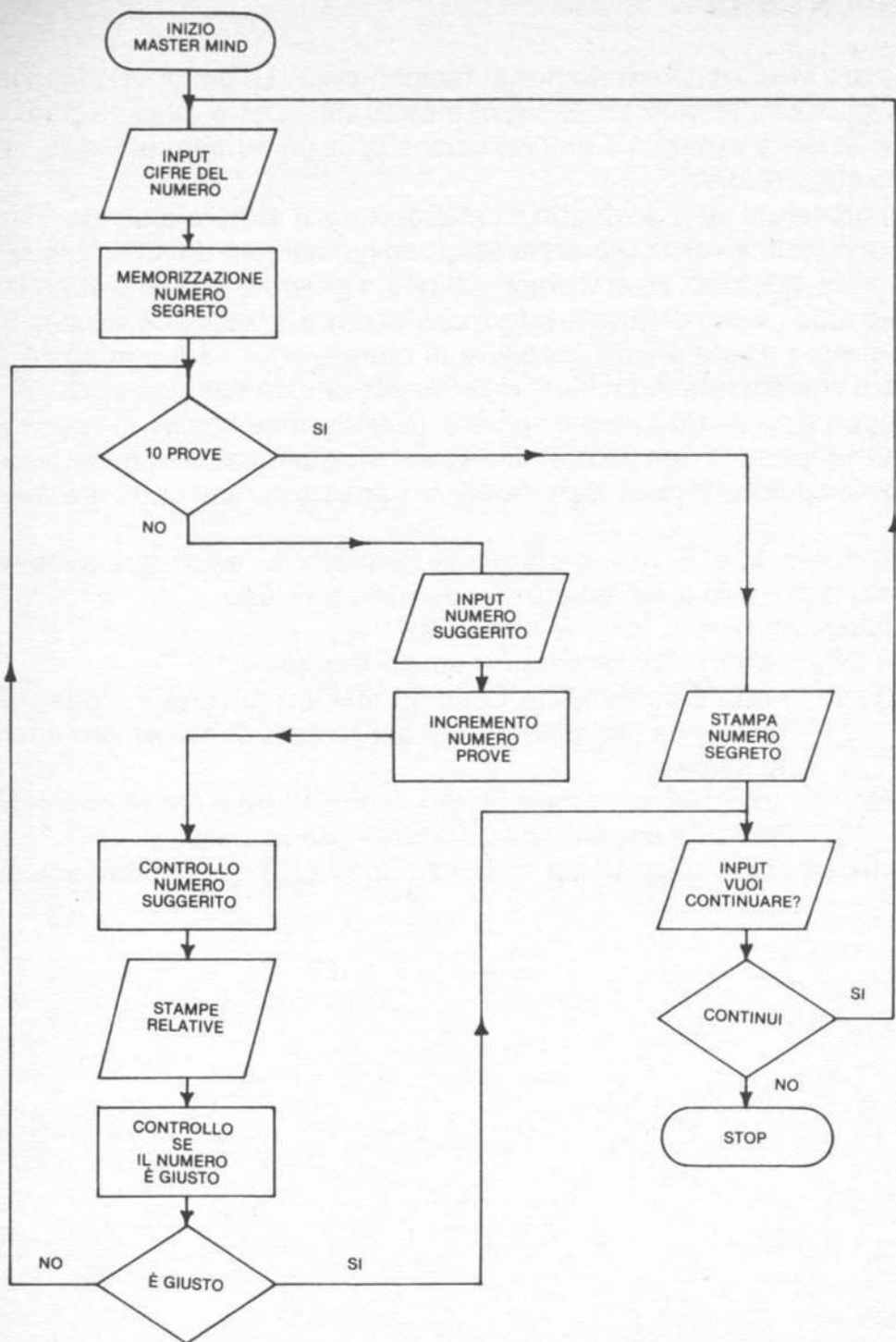
Voi proverete ad indovinarlo e l'elaboratore vi aiuterà dicendovi solo se le cifre che voi dichiarerete esistono nel numero da lui tenuto segreto e, se esistono, se si trovano nel posto giusto o in quello sbagliato. Facendo tesoro di queste informazioni potrete individuare il numero segreto. Potrete anche scegliere di quante cifre sarà composto il numero che dovrete indovinare e se vorrete che compaiano cifre ripetute oppure no all'interno del numero. Quest'ultima opzione si può attivare togliendo la riga 250 del listing del programma (com'è impostato ora non permetterà all'elaboratore di pensare numeri con cifre ripetute).

Con due cicli di FOR il programma controllerà l'esattezza o meno del vostro numero e poi emetterà i risultati su video.

Le strutture di dati sono le seguenti:

- A (N): vettore contenente il numero incognito;
- B (N): vettore contenente i numeri che voi di volta in volta immetterete nell'elaboratore per tentare di indovinare quello segreto;
- E$ (N): che contiene i risultati dei confronti tra il numero pensato dall'elaboratore e il numero pensato da voi.

Vi diamo ora il diagramma di flusso con la relativa traduzione in BASIC.

INIZIO
MASTER MIND
INPUT
CIFRE DEL
NUMERO
MEMORIZZAZIONE
NUMERO
SEGRETO
10 PROVE
SI
NO
INPUT
NUMERO
SUGGERITO
INCREMENTO
NUMERO
PROVE
STAMPA
NUMERO
SEGRETO
CONTROLLO
NUMERO
SUGGERITO
INPUT
VUOI
CONTINUARE?
STAMPE
RELATIVE
CONTINUI
SI
NO
STOP
CONTROLLO
SE
IL NUMERO
È GIUSTO
NO
È GIUSTO
SI

```
10  REM M A S T E R  M I N D
20  REM
30  REM VARIABILI USATE
42  REM  A(N)  VETTORE CON NUMERO INCOGNITO
44  REM  B(N)  VETTORE CON NUMERI TENTATIVI
46  REM
47  CLS
48  PRINT "L'ELABORATORE RISPONDERA' AI VOSTRI TENTATIVI CON:"
50  PRINT "A  SE UNA CIFRA SI TROVA AL POSTO GIUSTO"
52  PRINT "B  SE UNA CIFRA ESISTE, MA IN UN POSTO DIVERSO"
53  PRINT "I NUMERI INCOGNITI SONO COMPOSTI TUTTI DA CIFRE DIVERSE!!!!"
54  PRINT "DIGITA  V  PER CONTINUARE"
60  INPUT W$
70  IF W$="V" THEN GOTO 100
80  GOTO 60
100 INPUT "DI QUANTE CIFRE VUOI CHE SIA COMPOSTO IL NUMERO DA TROV
ARE";N
130 DIM A(N): DIM B(N)
150 DIM E$(N)
170 REM
190 REM INIZIA LA LISTA CASUALE PER MEMORIZZARE IL NUMERO DA TROVA
RE
200 REM
205 LET T=0
210 LET A(1)=INT(9*RND(1)+1)
220 FOR K=2 TO N
230   LET A(K)=INT(9*RND(1)+0)
240     FOR J=1 TO K-1
250       IF A(K)=A(J) THEN LET K=K-1
260     NEXT J
270 NEXT K
280 GOTO 500
300 FOR W=1 TO N
305   LET E$(W)=""
310 NEXT W
315 REM INIZIO CONTROLLI
320 S=1
325 FOR H=1 TO N
330   FOR J=1 TO N
335     IF B(H)=A(J) THEN LET E$(S)=" B"
340   NEXT J
345   IF E$(S)=" B" THEN LET S=S+1
350 NEXT H
360 LET L=1
370 FOR P=1 TO N
380   IF B(P)=A(P) THEN LET E$(L)=" A"
390   IF E$(L)=" A" THEN LET L=L+1
400 NEXT P
410 GOSUB 600
450 REM
455 REM CONFRONTO SE IL NUMERO E' QUELLO GIUSTO
460 REM
465 LET M=0
470 FOR V=1 TO N
472   IF A(V)=B(V) THEN LET M=M+1
474 NEXT V
476 IF M=N THEN GOTO 485
480 GOTO 500
```

```
483 CLS
485 PRINT "BRAVO! HAI TROVATO IL NUMERO IN ";T;" MOSSE"
490 GOTO 800
500 IF T>10 THEN GOTO 700
505 PRINT
510 FOR U=1 TO N
520   INPUT A$
530   IF A$="" THEN GOTO 520
540   LET B(U)=VAL(A$)
550 NEXT U
560 LET T=T+1
570 GOTO 300
600 REM RISULTATO CONFRONTI
610 CLS
620 PRINT "IL RISULTATO DEI CONFRONTI E':"
630 FOR K=1 TO N
640   PRINT E$(K);
650 NEXT K
660 RETURN
700 REM
710 REM S T A M P E
720 REM
725 CLS
730 PRINT
740 PRINT "IL NUMERO ESATTO ERA:"
750 FOR K=1 TO N
760   PRINT A(K);
770 NEXT K
800 REM
810 REM RICHIESTA SE SI CONTINUA A GIOCARE
820 REM
825 CLS
830 PRINT "VUOI CONTINUARE A GIOCARE (S/N) ?"
840 INPUT W$
850 IF W$="S" THEN GOTO 47
860 END
```

## PAROLIAMO

Abbiamo ora un gioco famosissimo e conosciuto dalla maggior parte delle persone. Esso consiste nel richiedere all'elaboratore 10 lettere casuali (vocali o consonanti) e nel cercare, in un tempo stabilito, la parola più lunga che si può formare con tali lettere. L'elaboratore, oltre a tenervi il punteggio, vi fornirà le vocali o consonanti che gli chiederete. Poi attenderà un tempo stabilito, che voi dovrete calibrare, allungando od accorciando il falso ciclo di FOR contenuto tra le istruzioni 320 e 340.

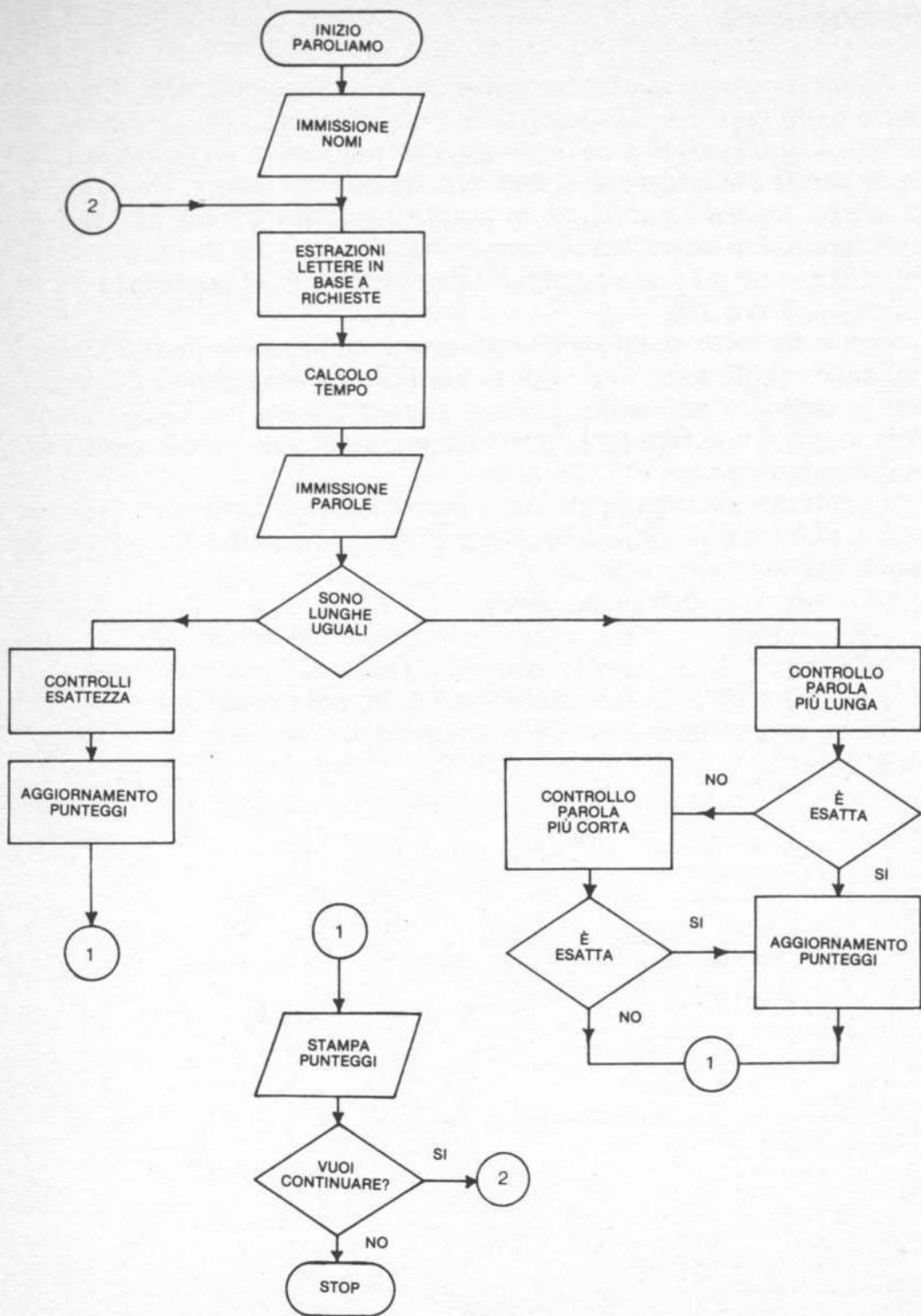
Una volta fatto questo e che gli avrete immesso le parole trovate dai concorrenti, esso farà i dovuti controlli ed assegnerà i punteggi. Poi vi chiederà se vorrete giocare ancora oppure no. Chiaramente l'elaboratore non farà un controllo semantico delle parole immesse, ma soltanto un controllo sintattico.

Il controllo del significato delle parole dovrete farlo voi in quanto non si può inserire l'intero vocabolario della lingua italiana nella memoria dell'elaboratore personale.

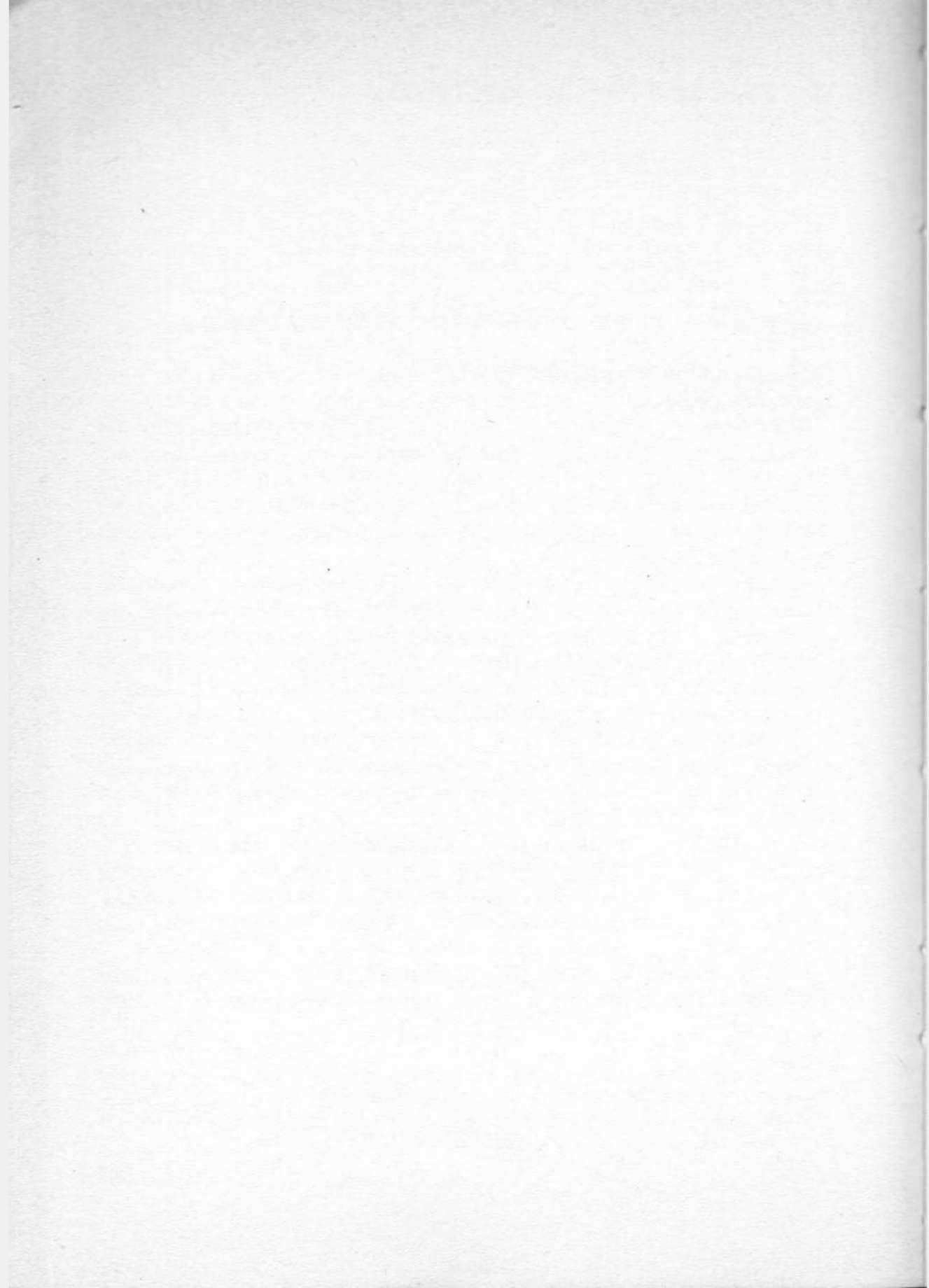
Le strutture di dati usate sono:

- PA$ e PB$: per le parole immesse dai concorrenti;
- Z$ (10): per le lettere proposte dall'elaboratore;
- A$ (10) e B$ (10): per contenere i nomi dei concorrenti.

Diamo ora, come al solito, il diagramma di flusso e la sua traduzione in BASIC.

INIZIO
PAROLIAMO
IMMISSIONE
NOMI
2
ESTRAZIONI
LETTERE IN
BASE A
RICHIESTE
CALCOLO
TEMPO
IMMISSIONE
PAROLE
SONO
LUNGHE
UGUALI
CONTROLLI
ESATTEZZA
AGGIORNAMENTO
PUNTEGGI
1
CONTROLLO
PAROLA
PIÙ LUNGA
NO
È
ESATTA
CONTROLLO
PAROLA
PIÙ CORTA
SI
È
ESATTA
SI
AGGIORNAMENTO
PUNTEGGI
NO
1
1
STAMPA
PUNTEGGI
VUOI
CONTINUARE?
SI
2
NO
STOP

```
10    REM
20    REM P A R O L I A M O
30    REM
40    REM VARIBILI USATE:
50    REM
60    REM A$,B$ PER I CONCORRENTI
70    REM PA$,PB$ PER LE PAROLE DEI CONCORRENTI
80    REM Z$ PER LE LETTERE PROPOSTE DAL GIOCO
90    REM
100   DIM PA$(10): DIM PB$(10): DIM C$(10): DIM D$(10): DIM Z$(10)
110   REM INSERIMENTO CONCORRENTI
120   CLS
130   INPUT "NOME PRIMO CONCORRENTE";A$
140   INPUT "NOME SECONDO CONCORRENTE";B$
150   REM ESTRAZIONE LETTERE
160   CLS
170   FOR I=1 TO 10
180     LET V=INT(42*RND(1)+1)
190     LET C=INT(76*RND(1)+43)
200     PRINT "VUOI VOCALE(VO) O CONSONANTE(CO)?"
210     INPUT R$
220     IF R$="" THEN 210
230     IF R$="VO" THEN X=V
240     IF R$="CO" THEN X=C
245     RESTORE
250     FOR J=1 TO X
260       READ B$
270     NEXT J
280     LET Z$(I)=B$
282     LET C$(I)=B$
284     LET D$(I)=B$
290     PRINT "LA LETTERA E': ";B$
300     CLS
310   NEXT I
312   REM TEMPO DI ATTESA: DA CALIBRARE!!!!
315   LET T=0
320   FOR K=1 TO 10000
330     LET T=T+1
340   NEXT K
350   CLS
360   REM IMMISSIONE PAROLE TROVATE
370   REM BISOGNA IMMETTERLE LETTERA PER LETTERA DANDO
380   REM UN RETURN OGNI LETTERA
390   PRINT "PAROLA PRIMO GIOCATORE:"
400   FOR G=1 TO 10
410     INPUT O$
420     LET PA$(G)=O$
430   NEXT G
435   PRINT "PAROLA SECONDO GIOCATORE:"
440   FOR G=1 TO 10
450     INPUT O$
460     LET PB$(G)=O$
470   NEXT G
480   REM CONTROLLO LUNGHEZZA PAROLE
490   LET CA=0: LET CB=0
500   FOR I=1 TO 10
510     IF PA$(I)<>"" THEN CA=CA+1
```

```
520  NEXT I
530  FOR I=1 TO 10
540    IF PB$(I)<>"" THEN CB=CB+1
550  NEXT I
560  IF CA<>CB THEN 700
570  GOSUB 3000
580  GOSUB 3500
590  IF FLGA=1 THEN PUNTA=PUNTA+CA
600  IF FLGB=1 THEN PUNTB=PUNTB+CB
610  GOTO 1000
700  IF CA<CB THEN GOTO 800
705  REM QUI PAROLA A MAGGIORE DI PAROLA B
710  GOSUB 3000
720  IF FLGA=0 THEN GOTO 750
730  LET PUNTA=PUNTA+CA
740  GOTO 1000
750  GOSUB 3500
760  IF FLGB=1 THEN PUNTB=PUNTB+CB
770  GOTO 1000
800  REM QUI PAROLA B MAGGIORE DI PAROLA A
810  GOSUB 3500
820  IF FLGB=0 THEN GOTO 850
830  LET PUNTB=PUNTB+CB
840  GOTO 1000
850  GOSUB 3000
860  IF FLGA=1 THEN PUNTA=PUNTA+CA
870  GOTO 1000
1000 REM CONTINUAZIONE OPPURE FINE
1010 CLS
1020 PRINT "A QUESTO PUNTO IL PUNTEGGIO E':"
1030 PRINT "CONCORRENTE: ";A$;" HA PUNTI: ";PUNTA
1040 PRINT "CONCORRENTE: ";B$;" HA PUNTI: ";PUNTB
1050 PRINT
1060 INPUT "VOLETE CONTINUARE A GIOCARE (S=SI,N=NO)";W$
1070 IF W$="" GOTO 1060
1080 IF W$="S" THEN GOTO 150
1090 STOP
2000 DATA I,I,I,I,I,I,I,I,I,I,I,I,O,O,O,O,O,
2010 DATA O,O,A,A,A,A,A,A,A,A,A,U,U,U,U,E,
2020 DATA E,E,E,E,E,E,E,B,B,C,C,C,C,C,C,C,D,
2030 DATA D,F,F,F,G,G,G,G,H,L,L,L,L,M,M,M,M,
2040 DATA M,M,N,N,N,N,N,N,P,P,P,P,P,P,Q,R,R,
2050 DATA R,R,R,R,R,R,S,S,S,S,S,S,S,S,S,S,T,
2060 DATA T,T,T,T,T,T,T,T,T,V,V,W,Z,Z,Z,Z
3000 REM CONTROLLO PAROLA A
3010 LET FLGA=0
3020 FOR K=1 TO CA
3025   LET J=1
3030   FOR J=1 TO 10
3040     IF PA$(K)<>C$(J) THEN GOTO 3070
3050     LET C$(J)="": LET PA$(K)=""
3060     GOTO 3080
3070   NEXT J
3080 NEXT K
3090 FOR K=1 TO 10
3100   IF PA$(K)<>"" THEN GOTO 3130
```

```
3110 NEXT K
3120 LET FLGA=1
3130 RETURN
3500 REM CONTROLLO PAROLA B
3510 LET FLGB=0
3520 FOR K=1 TO 10
3525   LET J=1
3530   FOR J=1 TO CB
3540     IF PB$(K)<>D$(J) THEN GOTO 3570
3550     LET D$(J)="": LET PB$(K)=""
3560     GOTO 3580
3570   NEXT J
3580 NEXT K
3590 FOR K=1 TO 10
3600   IF PB$(K)<>"" THEN GOTO 3630
3610 NEXT K
3620 LET FLGB=1
3630 RETURN
```

# 6. PROGRAMMI DI GRAFICA

## DISEGNO DI PUNTI IN UN PIANO CARTESIANO (*)

Il programma che segue vi darà la possibilità di disegnare dei punti in una determinata sezione del piano cartesiano.

Voi non dovrete far altro che fornire la sezione desiderata del piano cartesiano e le coppie di coordinate dei punti che vorrete rappresentare. Per disegnare il grafico di una funzione, dovrete fornire al programma più punti possibili del grafico e poi per interpolazione (o più semplicemente per unione di punti contigui), disegnare il vero e proprio grafico.

Questo programma, come anche il seguente, potrebbe essere utilizzato come routine di stampa per alcuni dati, provenienti da altri programmi e che a voi interessa visualizzare sul piano cartesiano.

L'algoritmo usato per tale scopo consiste nel memorizzare tutte le stampe su un'unica matrice, che potrebbe rappresentare il video, e poi stampare, riga per riga tutta la matrice.

Chiaramente la precisione del grafico non potrà essere massima in quanto, invece di disegnare una curva l'elaboratore disegnerà una serie di asterischi, ma ci potrà dare un'idea dell'andamento dei valori che sono al nostro studio.

Le strutture di dati fondamentali si possono così riassumere:

G (20,36): matrice su cui sono memorizzate le righe da stampare (chiaramente agendo sulla DIMENSION potremmo variare le dimensioni del grafico);

X (N) ed Y (N): contenenti le coordinate dei punti da disegnare.

Segue ora il diagramma di flusso con la relativa traduzione in BASIC.

(*) Il piano cartesiano è un piano usato nella geometria analitica per rappresentare diagrammi e funzioni che sono in sintesi un insieme di punti correlati tra loro. Per individuare il singolo punto si avrà bisogno, essendo un piano, di due coordinate: l'ascissa e l'ordinata.

INIZIO
GRAFICO
INPUT
DIMENSIONI
GRAFICO
INPUT
PUNTI DA
GRAFICARE
AZZERAMENTO
MATRICE
GRAFICA
CALCOLO
PUNTI FUORI
DAL GRAFICO
GENERAZIONE
MATRICE
GRAFICA
STAMPA
GRAFICO
STOP

```
10    REM
20    REM   GRAFICO DI PUNTI SU PIANO CARTESIANO
30    REM
40    REM  VARIABILI USATE:
50    REM
60    REM  G(20,36)    GRAFICO DA STAMPARE
70    REM  X(N),Y(N)  VETTORI CONTENENTI LE COORDINATE DEI PUNTI
80    REM
81    LET A=20:LET B=36
85    DIM G(A,B)
90    CLS
100   PRINT
110   PRINT "DAMMI IL LIMITE MINIMO DELLE ASCISSE:"
120   INPUT X1
130   IF X1<>INT(X1) THEN 120
140   PRINT "DAMMI IL LIMITE MASSIMO DELLE ASCISSE:"
150   INPUT X2
160   IF X2<>INT(X2) THEN 150
170   PRINT "DAMMI IL LIMITE MINIMO DELLE ORDINATE:"
180   INPUT Y1
190   IF Y1<>INT(Y1) THEN 180
200   PRINT "DAMMI IL LIMITE MASSIMO DELLE ORDINATE:"
210   INPUT Y2
220   IF Y2<>INT(Y2) THEN 210
230   CLS
240   INPUT "QUANTI PUNTI VUOI GRAFICARE";N
245   DIM X(N): DIM Y(N)
250   FOR I=1 TO N
260     PRINT "DAMMI L'ASCISSA DEL PUNTO ";I
270     INPUT X(I)
280     PRINT "DAMMI L'ORDINATA DEL PUNTO ";I
290     INPUT Y(I)
300   NEXT I
310   REM AZZERAMENTO MATRICE STAMPA
320   LET O=0
330   FOR I=1 TO A
340     FOR J=1 TO B
350       LET G(I,J)=0
360     NEXT J
370   NEXT I
380   REM RIEMPIMENTO MATRICE
390   FOR I=1 TO N
400     IF (X(I)<X1) OR (X(I)>X2) THEN 480
410     IF (Y(I)<Y1) OR (Y(I)>Y2) THEN 480
420     LET X(I)=X(I)/(X2-X1)
430     LET Y(I)=Y(I)/(Y2-Y1)
440     LET X(I)=INT(X(I)*(A-1)+1.5)
450     LET Y(I)=INT(Y(I)*(B-1)+1.5)
455     LET G(X(I),Y(I))=G(X(I),Y(I))+1
460     GOTO 490
470     REM
480     LET O=O+1
490     REM
500   NEXT I
510   REM S T A M P E
520   CLS
530   PRINT "PUNTI DA DISEGNARE:";N;"  INTERNI:";N-O;"   ESTERNI:";O
```

```
540  PRINT 'X DA ';X1;'A';X2;'DALL'ALTO IN BASSO'
550  PRINT 'Y DA ';Y1;'A';Y2;'DA SINISTRA A DESTRA'
560  PRINT
570  PRINT '* INDICA I PUNTI'
580  PRINT
590  PRINT 'I';
600  FOR J=1 TO B+1
610    PRINT '-';
620  NEXT J
630  PRINT 'Y'
640  FOR I=1 TO A
650    PRINT 'I';
660    FOR J=1 TO B
670      IF G(I,J)  0 THEN 710
680      PRINT ' ';
690      GOTO 720
700  REM
710      PRINT '*';
720  REM
730    NEXT J
740    PRINT
750  NEXT I
760  PRINT 'X'
770  REM FINE GRAFICO
780  END
```

## ISTOGRAMMA

L'istogramma è un particolare diagramma molto usato in statistica. Esso serve, avendo a disposizione parecchi numeri o dati, per vedere come essi si dispongono su una retta, vale a dire quanti di essi cadono in uno specifico intervallo di tale retta.

Le barre che comporranno l'istogramma saranno lunghe in modo proporzionale alla quantità di numeri contenuti nell'intervallo rappresentato dalla barra in esame.

Un tale tipo di analisi è, come già detto, molto utile in statistica, per poter vedere come si distribuiscono i rilevamenti campione. Ma anche in qualche altro caso di interesse meno scientifico, potremmo trarne giovamento.

Il problema per disegnare l'istogramma è il seguente: bisogna fare in modo che, supponendo che la somma della lunghezza di tutte le barre sia uno, la lunghezza di ciascuna barra stia in rapporto con la lunghezza di un'altra, nello stesso modo in cui il numero dei dati contenuti nel primo intervallo stia al numero dei dati contenuti nel secondo intervallo. Ciò si ottiene tramite delle normalizzazioni.

Quindi l'algoritmo usato consisterà soltanto nell'osservazione dell'intervallo di appartenenza dei vari numeri seguita poi da una normalizzazione per poterli stampare in modo coerente.

Le strutture di dati fondamentali sono in pratica:

- H (20): contenente la lunghezza di ogni barra dell'istogramma;
- X (200): contenente il valore dei numeri su cui conduciamo la nostra analisi.

Chiaramente si potrà, al solito, agire sui DIMENSION per variare sia il numero di barre dell'istogramma, sia il numero dei dati da analizzare.

Anche questo secondo programma potrà essere usato per analizzare dei dati provenienti da un diverso programma: bisognerà soltanto far diventare gli output del programma principale, gli input del programma di grafica.

Segue ora il diagramma di flusso con la sua traduzione in BASIC.

INIZIO
ISTOGRAMMA
ASSEGNAZIONI
COSTANTI
E LIMITI
ISTOGRAMMA
INPUT
ELEMENTI
LISTA
DISPOSIZIONE
DEGLI ELEMENTI
IN BARRE
NORMALIZZAZIONE
STAMPA
ISTOGRAMMA
STOP

```
10    REM
20    REM I S T O G R A M M A
30    REM
40    REM VARIABILI USATE:
50    REM  H(20) PER CONTENERE LA LUNGHEZZA DELLE BARRE DELL'ISTOGR
AMMA
60    REM  X(200) PER CONTENERE I NUMERI DA ANALIZZARE
70    REM  X1 ED X2  LIMITI DELL'ISTOGRAMMA
80    REM
90    REM  DEFINIZIONI COSTANTI
100   LET HI=20
110   LET N=200
120   LET LNORM=10
130   REM DIMENSIONI
140   DIM H(HI): DIM X(N)
150   CLS
160   REM INTRODUZIONE DATI
170   PRINT
180   PRINT "DAMMI IL LIMITE MINIMO DELL'ISTOGRAMMA:"
190   INPUT X1
200   IF INT(X1)<>X1 THEN 190
210   PRINT "DAMMI IL LIMITE MASSIMO DELL'ISTOGRAMMA:"
220   INPUT X2
230   IF INT(X2)<>X2 THEN 220
240   REM
250   FOR I=1 TO N
260     PRINT "DAMMI IL NUMERO ";I;" DELLA LISTA DA ANALIZZARE"
270     INPUT X(I)
280     IF INT(X(I))<>X(I) THEN 270
290   NEXT I
300   REM
310   REM COSTRUZIONE DELLE BARRE DELL'ISTOGRAMMA
320   REM
330   REM INIZIALIZZAZIONE
340   LET I=(X2-X1)/HI
350   LET M=LNORM
360   REM ESECUZIONE
370   FOR I=1 TO N
380     LET K=INT(HI*X(I)/(X2-X1)+1)
390     LET H(K)=H(K)+1
400     IF H(K) =M THEN GOTO 420
410     LET M=H(K)
420   NEXT I
430   REM
440   REM STAMPE ISTOGRAMMA
450   REM
460   CLS
470   PRINT
480   PRINT "ISTOGRAMMA DI ";N;" DATI"
490   PRINT "DA ";X1;" A ";X2;" AD INTERVALLI DI: ";I
500   PRINT "CON LA MASSIMA ALTEZZA POSSIBILE UGUALE A ";M;" PUNTI"
510   PRINT "OGNI PUNTO RAPPRESENTA "; M/LNORM;" UNITA'"
520   REM STAMPA
530   FOR I=1 TO HI
540     PRINT "I";
550     IF H(I)=0 THEN GOTO 590
```

```
560    FOR J=1 TO INT(H(I)/M*LNORM+0.5)
570      PRINT '*';
580    NEXT J
590    PRINT
600  NEXT I
610  REM FINE STAMPE
620  END
```

# 7. COME ADATTARE I PROGRAMMI AL PROPRIO PERSONAL COMPUTER

In questo capitolo daremo un rapido sguardo ad un argomento che non è molto trattato a livello teorico, ma che nella pratica assume un posto di rilievo: ossia l'adattamento di programmi studiati per elaboratori diversi da quello che voi sarete costretti o vorrete usare.

Questo problema è di soluzione abbastanza complessa, perché implica conoscenze diverse. In primo luogo si dovrebbe avere una discreta dimestichezza con le tecniche di programmazione per poter capire ogni tipo di programma.

In secondo, bisognerebbe anche conoscere molto bene le caratteristiche dei due elaboratori in gioco: vale a dire quello da cui si preleva il programma e quello su cui lo si vuole adattare.

Tuttavia, qui vedremo che, anche non essendo a conoscenza di tutti questi elementi, si potrà ottenere qualche risultato.

## 7.1 DIFFERENZE FRA ELABORATORI DI TIPO DIVERSO

Cominciamo con il vedere le differenze che possono esistere tra elaboratori di tipo diverso, senza dimenticare che stiamo trattando elaboratori di una stessa fascia, tra quelle definite nel capitolo primo.

In particolare ci riferiremo agli home computer.

Innanzitutto vediamo il linguaggio di programmazione. Due elaboratori aventi il BASIC come linguaggio di programmazione, non possono avere, a livello di istruzioni BASIC, molte differenze tra di loro. Voglio dire che la semantica di una istruzione sarà sicuramente la stessa sia in una macchina che in un'altra.

Potrà tutt'al più cambiare la sintassi di un'istruzione: vale a dire ci sarà bisogno di qualche segno di punteggiatura o di qualche spazio, ma l'insieme delle istruzioni non varierà. Sarà solamente compito vostro scorrere il manuale fornito dalla casa costruttrice per vedere dove ci sono delle diversità di sintassi o meno.

Grosse differenze potranno, invece, esistere a livello di sistema operativo delle macchine, vale a dire a livello dei programmi che gestiscono l'elaboratore nel suo funzionamento (nel sistema operativo sono comprese tutte le utilità a disposizione della macchina come copia di file, lista dei contenuti della memoria etc.).

Comunque questo lato particolare dell'elaboratore, non ci toccherà per nulla, per quel che riguarda la programmazione. Infatti quando si lavora in BASIC ci si può dimenticare tutto quello che vi è intorno ed agire, anche se con il paraocchi, all'interno di esso.

Un'ultimo lato da evidenziare, è quello relativo all'input e all'output, vale a dire all'immissione e alla scrittura di dati e di risultati. Dobbiamo fondamentalmente capire che ogni elaboratore è diverso dall'altro per quel che riguarda questa caratteristica. Se noi dovessimo emettere dei risultati su una stampante, su video o su un file troveremmo che ogni elaboratore ha le sue istruzioni con tutti i problemi connessi.

Ricordo che ho trasportato diversi programmi da un tipo di elaboratore ad un'altro, e la maggior parte delle volte, i problemi li riscontravo proprio in queste due operazioni fondamentali di ingresso e uscita dei dati. Chiaramente se ci interessa avere soltanto dei risultati su video e niente altro, a parte il numero di righe e colonne, sempre poste sotto controllo, non dovremmo incontrare grosse difficoltà. Ma volendo avere dei risultati su file, per non parlare della stampante, i problemi potrebbero moltiplicarsi.

I programmi precedentemente illustrati funzionano sull'elaboratore SINCLAIR SPECTRUM.

Vedremo nei prossimi due paragrafi i provvedimenti che si potranno prendere, senza avere conoscenze approfondite dell'hardware di altri elaboratori, per farli funzionare su di essi.

## 7.2 COME ADATTARE I PROGRAMMI

Analizziamo qui di seguito come, solo conoscendo il BASIC delle macchine in oggetto, si possano adattare dei programmi, scritti per un elaboratore, ad un altro elaboratore.

Di cosa ci serviremo? Quali saranno i passi logici e fisici che dovremmo compiere?

Se il programma in oggetto è scritto su un dischetto o inciso su un nastro, la prima cosa che si dovrà fare sarà quella di effettuare una copia su un altro dischetto o su un altro nastro del programma in og-

getto. Tutte le modificazioni saranno poi effettuate sulla copia del programma lasciando così intatto l'originale. Ciò per due buoni motivi. Il primo perché, quasi sicuramente, il programma ci sarà stato fornito da un amico e perciò richiederà indietro l'originale. Il secondo motivo è, invece, il seguente: se dopo aver fatto alcune modifiche al programma ci accorgiamo di aver sbagliato parecchie cose o di trovare un metodo più rapido di conversione, invece di ricorreggere il programma già modificato, potremo richiamare nuovamente il programma dal dischetto originale, cancellando quello sbagliato.

Ciò che abbiamo appena espresso è il problema del *Backup*. Quasi nessun libro di programmazione pone il dovuto accento su questa problematica. Eppure è una cosa fondamentale avere sempre a disposizione un dischetto, ben conservato, o un nastro contenente tutti i programmi da noi prodotti e funzionanti. Non dimenticatevi mai che una cosa molto semplice da fare, sugli home computer, è la cancellazione di file o di programmi utili! Quindi l'avere sempre una copia di ciò che ci interessa, gelosamente conservata, non è un'idea così inutile.

Ma torniamo al nostro problema di adattare i programmi.

Il secondo passo da fare è il seguente. Entrare nella logica del programma. Vale a dire sviscerare in ogni passo logico il programma da convertire. Isolare, tra di loro, le istruzioni di commento, le istruzioni di input e/o output, le istruzioni di elaborazione vera e propria, le istruzioni di memorizzazione dati e così via. Tutto ciò avrà chiaramente uno scopo, oltre a farvi conoscere il programma più da vicino. Infatti il fine di questo secondo processo sarà la traduzione del programma: indovinate su cosa? Chiaramente in diagramma di flusso (se non siete così fortunati da averlo già a disposizione tramite il vostro amico che vi ha prestato il programma).

Ecco un ulteriore motivo per cui abbiamo introdotto i diagrammi di flusso. Essi sono molto utili in questa fase di adattamento dei programmi.

Avendo ora a disposizione il diagramma di flusso complessivo del programma, non vi rimarrà che da eseguire l'ultimo passo: tradurre il diagramma nel linguaggio BASIC del nuovo elaboratore.

A questo punto i più inesperti si potranno domandare: per quale motivo ci siamo fatti prestare il programma se poi lo abbiamo dovuto riscrivere personalmente tramite il diagramma di flusso?

Ebbene il trucco è questo: non dovremo riscrivere tutto il programma, ma solo una sua piccola parte, che potrà essere quantificata intorno al 10 o 20% di istruzioni. Infatti come abbiamo già detto i BASIC

delle due macchine non potranno differire di molto, ma i più grossi problemi si troveranno nelle istruzioni di input o output e in alcune regole sintattiche. Perciò lasciando inalterato quasi tutto il programma, andremo a modificare, e grazie al diagramma di flusso andremo quasi a colpo sicuro, soltanto quelle poche istruzioni che saranno differenti tra le due macchine.

Chiaramente dopo aver preso un po' di pratica, specialmente con il proprio elaboratore, si potrà anche fare a meno di costruire il diagramma di flusso e si andranno direttamente a modificare le parti di programma che riconosceremo avere sicuramente dei problemi.

Vi sarà anche, per chi già possiede una buona conoscenza del proprio elaboratore, un diverso modo di procedere. Si fa girare una prima volta il programma sul proprio elaboratore e poi si osservano i diagnostici di errore che darà il BASIC. Se tali diagnostici sono abbastanza documentati, vale a dire se possiedono almeno l'indicazione dell'istruzione che non è stata bene interpretata dall'elaboratore, avremo tutti gli elementi per risalire all'istruzione o al gruppo di istruzioni da cambiare.

Nel prossimo paragrafo saranno elencate e spiegate alcune modifiche necessarie ai programmi precedentemente descritti, per poterli far girare su elaboratori diversi dal SINCLAIR SPECTRUM per cui sono stati originariamente costruiti.

## 7.3. QUALCHE ESEMPIO DI ADATTAMENTO

Vedremo ora le correzioni che dovrete apportare ai programmi precedentemente descritti per poterli far funzionare su altri due tipi di elaboratori fra i più diffusi in commercio: il COMMODORE e l'APPLE. Tali correzioni riguardano alcune istruzioni BASIC che differiscono tra i tre tipi di elaboratore prima menzionati (il terzo è chiaramente lo SPECTRUM SINCLAIR su cui i programmi già girano).

La prima modifica riguarda l'istruzione BASIC che serve per pulire il video. Come già sappiamo, per il SINCLAIR si usa l'istruzione:

CLS

Per il COMMODORE, invece, ogni volta che dovrete pulire il video dovrete usare l'istruzione:

PRINT «CLEAR»

Per l'APPLE vi sarà una diversa istruzione utile allo scopo, vale a dire:

HOME

Un'altra istruzione che ha peculiarità diverse tra i tre tipi di elaboratori è l'istruzione:

LET

Mentre nel SINCLAIR occorre usarla ogni volta che si fa un'assegnazione, per gli altri due tipi di elaboratore si può anche fare a meno di usarla.

Un'ulteriore differenza tra i tre tipi di elaboratori che stiamo esaminando, consiste nella ricerca di stringhe in un vettore. Come abbiamo visto, nel SINCLAIR si è dovuto usare l'artificio di scrivere le stringhe su delle matrici e poi di ricercarle come se fossero scritte in vettori. Negli altri tipi di elaboratori tutto ciò non serve. In effetti basterà memorizzare le stringhe in vettori e poi ricercarle come è stato fatto sul SINCLAIR. Perciò per far girare i programmi che usano la ricerca di stringhe in matrici, basterà cambiare i DIMENSION delle matrici contenenti le stringhe e trasformarle in vettori.

Ad esempio per il programma dell'elenco telefonico, basterà cambiare le righe da 60 a 110 in questo modo:

```
60 DIM C$(135)
70 DIM N$(135)
80 DIM I$(135)
90 DIM A$(135)
100 DIM T$(135)
110 DIM P$(135)
```

Un'ultima cosa da sottolineare riguarda l'istruzione:

GOSUB Z*1000

dove Z è una variabile.

Mentre sul SINCALIR tale istruzione gira, sugli altri elaboratori non esiste un'istruzione che dica di andare ad una subroutine variabile. Perciò dovremo trovare un'altra istruzione che faccia al caso nostro. Tale istruzione è la:

ON...GOSUB...

in cui al posto dei primi puntini andrà inserita la variabile che controlla la subroutine a cui bisogna arrivare, e al posto dei secondi le righe dove si trovano le varie subroutine.

Ad esempio, sempre per il programma di gestione dell'elenco telefonico, al posto dell'istruzione di riga 280 andrà messa, negli altri tipi di elaboratore, l'istruzione seguente:

```
ON Z GOSUB 1000,2000,3000,4000,5000,6000
```

Così, in tutti gli altri casi in cui è stata usata una GOSUB variabile, dovrete effettuare delle sostituzioni simili a quella appena illustrata, quando vorrete usare i programmi su COMMODORE o APPLE.